APPENDIX I

MediSyn COM Interfaces

DOCKET ALARM

MediSyn COM Interfaces

Overview:

The following is a list of the interfaces the COM component will support. Each of the following interface definitions is composed of the name, followed by the method description, the signature, the parameters and the return type. The table of contents lists each interface followed by their methods. Each method will have a brief description. The names of the methods within the table of contents are hyperlinks to the method description detail.

The description will describe the functionality of the method. This will include any constraints or preprocessing information necessary to call this method.

The signature, for the purposes of this document, is written using MS Visual Basic syntax. This facilitates readability for the developers on this project, as the middle tier component will be developed using MS Visual Basic.

The parameter's names each parameter in the method call and gives a brief description of the purpose of the parameter. If the parameter has specific domain constraints, they will be noted in an indented small font beneath the parameter description.

The return type will be consistent for each method. The return data type will be named, followed by the rules of what data will be returned.

***note: All ADODB. Recordsets may be handled as 2 dimensional arrays, rather than a recordsest.*

Table of Contents

getCodeTable - Returns the specified code table. Used to populate dropdown lists.

IProblem_Drugs

getDrues - Returns list of drugs.
updateDrugs - Updates prescriptions, including start date and end date and active flag.
getProblems - Returns list of problems.
updateProblems - Updates problems, including active flag.

IEntity

getEntity - Returns the specified entity's record.
updateEntity - Updates the specified entity's record.
getPatientInsurance - Returns patient insurance records.
updatePatientInsurance- Updates the patient's insurance information.

IContact

getAddressInfo - Returns a list of address records for an entity.
updateAddressInfo - Updates an entity's address record.
getContactInfo - returns a list of contact records for an entity.
updateContactInfo - Updates an entity's contact info record.

IAccounting

bill - Creates an AR records.
getBills - Generates bills from AR records and returns a list of bill records.
getAcctsPavable - Returns a list of AP Records.
proceessChecks - Creates check records based on AP records.
processReceipts - Creates receipt records which are applied to existing bill records.
getReceipts - Returns a list of receipt records.
updateDeposit - Creates a deposit record based on existing receipt records.
getReconciliation - Returns deposit and check records that have not cleared.
updateReconciliation- Updates deposit and check records that have cleared.

COM Methods

getApplicants

Description: This method will return the specified applicant or all applicants in the applicant's table. The number of records returned will be limited to 20. This will ensure that the web interface is not overwhelmed with hundreds of records.

Signature:

```
Public Function getApplicants(rsApplicants as ADODB.Recordset,
                    Optional iApplicantId as Integer) as Long
```

Parameters:
rsApplicants: Contains applicant records.
iApplicantId: specifies the applicant to be returned.

The following columns are returned: Applicant Id, Entity Type, FName, LName, OrgName, UniqueId, FEI_SSN, Phone, Fax, Email, Entity Type, UserName, Password.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

updateApplicants

Description:
This method will create a MediSyn user from an applicant and flag that applicant as accepted, or flag the applicant as denied if rejected. Once an applicant has been passed to updateApplicant their record in the applicant table is flagged as approved or denied and the date of the transaction is recorded. 60 days after this date, the record will be removed. The record will stay in the database for 60 days for auditing purposes. After that time, any issues with the applicant should have been resolved

Signature:
Public Function updateApplicants(iApplicantId() as Integer, _
Optional isApproved() as Boolean) as Long

Parameters:
iApplicantId(): Array, specifies the applicant to be updated. The isApproved flag will determine whether the applicant becomes a member of MediSyn.
isApprovedo: If true a new MediSyn user is created and the applicant record is flagged as approved. If false the applicant record is flagged as denied.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

newApplicant

Description:
This method will create a new applicant and if the isApproved optional parameter = true, a new user. All of the parameters with valid data will be used to populate the applicant table and if the applicant is approved, the MediSynUser table, the corresponding entity table as specified by iEntityType, and the Contact_Info table.

Signature:
Public Function newApplicant(sFName as String, _
sLName as String, _
sOrgName as String,
sUniqueId as String,
sFEI-SSN as String, -

sPhone as String, _
sFax as String, _
sEMail as String, _
iEntityType as Integer, _
sUserName as String, _
sPassword as String, _
Optional iReferrer as Integer, _
Optional isApproved as Boolean) as Long

Parameters:
sFName : Specifies the first name if the applicant is a doctor.
sLName : Specifies the last name if the applicant is a doctor.
sOrgName : Specifies the organization name if the applicant is not a doctor.
sUniqueId : Specifies the unique identifier for the entity being added. UPIN for doctor, license forPharmacy.
sFEI SSN : Specifies the SSN or FEI for the entity being added.
sPhone : Specifies the phone number for the entity being added.
sFax : Specifies the fax number for the entity being added.
sEMail : Specifies the email address for the entity being added.
iEntityType : Specifies the entity type for the entity being added.
sUserName : Specifies the user name for the entity being added.
sPassword : Specifies the password for the entity being added.
iReferrer : Specifies the doctor _id of the doctor who will be the parent id for the doctor being added.
isApproved: If true a new MediSyn user is created and the applicant record is deleted. If false the applicant record is deleted.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

login

Description:
The login method provides a mechanism for MediSyn users to access the application.

Signature:
Public Function login(sLogin as String, _
sPassword as String, _
sIP_Address as String, _
ByRef iEntityType as Integer, _
ByRef iEntityId as Integer) as Long

Parameters:
sLogin: Specifies the username of the applicant. This value cannot contain <space> characters and must be a minimum of 6 and a maximum of 15 characters in length.
sPassword: Specifies the password of the applicant. This value cannot contain <space> characters and must be a minimum of 6 and a maximum of 15 characters in length. It

must also contain at least one uppercase and one lowercase letter and at least one numeric character.

sIP_Address: Specifies the IP Address of the user attempting to login.
iEntityType: The type of entity of the user logging in. This is returned to the caller.
iEntityId: The id of entity of the user logging in. This is returned to the caller.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

logout

Description:
The logout method provides a mechanism to inform MediSyn that a user has left the application. This is used for logging purposes. By noting the date, time and location of the login attempt, the application can track when and what lengths of time users have used the system.

Signature:
```
Public Function logout(ByVal iEntityId as Integer, _
                       ByVal iEntityType as Integer ) as Long
```

Parameters:
iEntityId: Specifies the id of the entity logging out.
iEntityType: Specifies the type of user logging out.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getUser

Description:
This method will return the specified user from the MediSynUser table. Administrators will use this method when a user forgets their login or password and they do not have an email address.

Signature:
```
Public Function getUser(ByVal iEntityType as Integer, _
                        sUniqueId as String, _
                        sLogin as String, _
                        sPassword as String ) as Long
```

Parameters:
iEntityType: Specifies the entity whose info is to be sent
sUniqueId: Specifies the unique identifier for the type of user defined by iEntityType above.
Examples: Doctor - UPIN, Pharmacy - License, FEI - Hospital.
sLogin - Where the user's login name is returned.
sPassword - Where the user's login name is returned.

Return type:

Long: returns 0 if successful and an Error Code if unsuccessful.

updateUser

Description:
This method updates user info to allow the user to change their login name and/or password.

Signature:
```
Public Function updateUser(ByVal iEntityId as Integer, _
                           ByVal iEntityType as Integer, _
                           Optional sUserName as String, _
                           Optional sPassword as String ) as Long
```

Parameters:
iEntityId: Specifies the entity whose info is to be updated.
iEntityType: Specifies the entity whose info is to be updated
sUserName: Specifies the new username
sPassword: Specifies the new password.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

sendUserInfo

Description:
This method gets user info and sends the user their login name and password via email. This provides the same functionality as getUser, except that it does not require the administrator's involvement.

Signature:
```
Public Function sendUserInfo(ByVal iEntityType as Integer, _
                             sUniqueId as String ) as Long
```

Parameters:
iEntityType: Specifies the entity whose info is to be sent
sUniqueId: Specifies the unique identifier for the type of user defined by iEntityType above.
Examples: Doctor - UPIN, Pharmacy - License, FEI - Hospital.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getNotes

Description:
This method allows the interface to receive all of the progress notes for a physician, patient, drug, or problem. The collection of notes returned is determined by the iNoteType parameter. This value will resolve to referred notes, open notes, problem notes, or drug notes.

Signature:

```
Public Function getNotes(ByVal iEntityId as Integer, _
                         iNoteType as Integer,
                         rsEntityNotes as ADODB_Recordset, _
                         Optional iDrug_Problem as Integer ) as Long
```

Parameters:

iEntityId: Specifies the doctor who is logged into the system. If iNoteType = 1 or 2, the iEntityId points to doctor _id. If iNoteType = 3 or 4, the iEnftyId points to patient id

iNoteType: Specifies the progress notes to return.

1 -referred notes
2 - open notes
3 - drug notes
4 - problem notes

rsReferringNotes: An ADO Recordset which will contain all of the progress notes where iDoctorId is the referring doctor id.

iDrug Problem: This field will contain the drug id or problem id if the iNoteType is 3 or 4 respectively.

The following columns are returned: note_id, patient.first_name, patient.last_name, note_date, note_time, ctor.first_name, doctor.last_name

Return type:

Long: returns 0 if successful and an Error Code if unsuccessful.

<u>getProgressNote</u>

Description:

The two dimensional array or ADODB.Recordset, rsProgressNote, returned will contain the fields in the progress note identified by iNoteId. The two dimensional array or ADODB.Recordset, rsDrugs, returned will contain the drug records associated with the progress notes identified. The two dimensional array or ADODB.Recordset, rsProblem, returned will contain the problems associated with the progress note identified.

Signature:

```
Public Function getProgressNote(ByVal iNoteId as Integer, _
                                rsProgressNote as ADODB.Recordset, _
                                rsDrugs as ADODB.Recordset, _
                                rsProblems as ADODB.Recordset ) as Long
```

Parameters:

iNoteId: Specifies the patient for whom the demographic information is requested.

rsProgressNote: An ADO Recordset which will contain the referenced progress note.

The following columns are returned: note_id, note_date, note_time, doctor_id, note, referring_Doctor, valid, transcript_id.

rsDrugs: An ADO Recordset which will contain those drugs referenced in the progress note.

The following columns are returned: drug_id, description, isActive.

rsProblems: An ADO Recordset which will contain those problems referenced in the the referenced progress note.

The following columns are returned: problem_id, description, isActive.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

updateProgressNote

Description:
Adds or updates a progress note for an existing patient and doctor. If the iNoteId parameter is 0, then the progress note is coming from the transcription service and is to be inserted into the Progress _Note table. If the iNoteId parameter > 0, then the note is being validated by the doctor and will be updated with the problems and drugs associated with the note. Also, a progress note may only be updated once. At that time the valid flag is set to true and the record can no longer be modified.

Signature:
Public Function updateProgressNote (ByVal iPatientId as Integer,
ByVal iNoteId as Integer,
ByVal iDoctorId as Integer,
ByVal iTranscriptionId as Integer,
sNoteDate as String,
sNoteText as String,
Optional aProblems() as Variant,
Optional aDrugs() as Variant,
Optional sNoteTime as String,
Optional ByVal iRefDoctorId as Integer) as Long

Parameters:
iPatientId: Specifies the patient referenced in the progress note.
iNoteId: Specifies the note id of the progress note. If this value is zero, then a new progress note is created.
iDoctorId: Specifies the doctor referenced in the progress note.
iTranscriptionId: Specifies the transcription service referenced in the progress note.
sNoteDate: Specifies the date of the progress note.
sNoteText: Specifies the text of the progress note.
aProblems: Two-dimensional array specifying the problems in the note. The columns are problemId, description, active. If the problemId is 0, then the problem is inserted into the problem table. All problems should be tagged to the progress note for future reference.

aDrugs: Two-dimensional array specifying the Drugs in the note. The columns are DrugId, description, active. If the DrugId is 0, then the drug is inserted into the drugs table. This array does not have to contain data. There may or may not be a prescription associated with a progress note. All drugs should be tagged to the progress note for future reference.
sNoteTime: Specifies the time of the progress note.
iRefDoctorId: Specifies the referring physician for the progress note.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

findPatient

Description:
The two dimensional array or ADODB.Recordset returned will contain the patient's name, SSN and DOB. This will be presented to. the user as a hyperlink to view the patient's problem(diagnosis) and drug(prescription) lists. At least one of the fields is required to have a value.

Signature:
Public Function findPatient(rsPatients as ADODB. Recordset,
Optional sFName as String, _
Optional SLName as String, _
Optional sSSN as String, _
Optional sDOB as String) as Long

Parameters:
rsPatients: An ADO Recordset which will contain all of the patient records that matched the search criteria.
sFName: Specifies the first name of the patient being searched.
sLName: Specifies the last name of the patient being searched.
sSSN: Specifies the SSN of the patient being searched.
sDOB: Specifies the DOB of the patient being searched.

The following columns are returned: patient_id, patient.first_name, patient.last_name, patient_ssn,, patient_dob, doctor_id.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

search

Description:
The two-dimensional array or ADODB.Recordset returned, will contain a list of all records that matched the search criteria.

Signature:
Public Function search(ByVal iEntityType as Integer, _

rsSearch as ADODB.Recordset
Optional sName as String, _
Optional sFName as String, _
Optional sLName as String, _
Optional sCity as String, _
Optional sZip as String) as Long

Parameters:
iEntityType: Specifies the type of entity to be searched. shame: Specifies the organization name of the entity to be searched. This valid when searching on any entity except doctors.
sFName: Specifies the first name of the entity to be searched. This is only valid when searching for doctors.
sLName: Specifies the last name of the entity to be searched. This is only valid when searching for doctors.
sCity: Specifies the city of the entity to be searched.
sZip: Specifies the zip code of the entity to be searched.
rsSearch: An ADO Recordset which will contain all matches for the search.

The following columns are returned: name, entity_id, entity_type, fNarne, LName, City, StateCode, zip.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getErrorMsg

Description:
This method will select the Error Message from the database and return it to the calling program. The return value is a string; which can be displayed to the user in the form of a message/alert window.

Signature:
Public Function getErrorMsg(ByVal lErrorCode, _
rsErrorMsg as ADODB.Recordset) as Long

Parameters:
lErrorCode: Specifies the error in the database.
rsErrorMsg: Recordset to contain the returned error data.

The following columns are returned: code_id, code_description

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getCodeTable

Description:

This method will return all records from the specified code table.

Signature:
Public Function getCodeTable(sCodeTable as String, _
rsCodeTable as ADODB.Recordset)

Parameters:
sCodeTable: Specifies the code table to be returned.
rsCodeTable: Specifies a recordset containing the ids and names of the fields in the code table specified.

The following columns are returned: Field_Id, Field_Name.

Return type:
Integer: returns positive if successful and 0 or negative if unsuccessful.

getDrugs

Description:
The two dimensional array or ADODB.Recordset returned will contain the specified entity's drug list.

Signature:
Public Function getDrugs(ByVal iEnityId as Integer, _
ByVal iEntityType as Integer, _
rsDrugs as ADODB.Recordset) as Long

Parameters:
iEntityId: Specifies the entity for whom the drug list is requested.
iEntityTYpe: Specifies the entity type of the entity for whom the drug list is requested.
rsDrugs: An ADO Recordset which will contain all of the prescription records linked to a patient.

The following columns are returned: prescription_id, description, doctor_id, isActive.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

updateDrugs

Description:
This method updates the progress_prescription table with the start_date, end_date.

Signature:
Public Function updateDrugs(ByVal iPrescriptionId as Integer, _
sStartDate as String, _
sEndDate as String
Optional ByVal iPharmacistId as Integer, -

Optional isActive as Boolean) as Long

Parameters:
iPrescriptionId: Specifies the drug record being updated
sStartDate: Specifies the prescription starting date.
sEndDate: Specifies the prescription ending date.
iPharmacyId: Specifies the pharmacy for whom the drug list is requested.
isActive: Specifies whether the drug is actively used.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getProblems
The two dimensional array or ADODB.Recordset returned will contain the specified entity's problem(diagnosis) list.

Signature:
Public Function getProblems(ByVal iEntityId as Integer, _
ByVal iEntityType as Integer, _
rsProblems as ADODB.Recordset) as Long

Parameters:
iEntityId: Specifies the enitity for whom the problem list is requested.
iEntityTYpe: Specifies the type of the entity for whom the problem list is requested.
rsProblems: An ADO Recordset which will contain all of the problem records linked to a patient.

The following columns are returned: problem_id, description, doctor_id, isActive.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

updateProblems

Description:
This method updates the progress_problem table with the active flag.

Signature:
Public Function updateProblem(ByVal iProblemId as Integer, _
isActive as Boolean) as Long

Parameters:
iProblemId: Specifies the problem being updated.
isActive: Determines whether the problem is actively used.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getEntitv

Description:
The two dimensional array or ADODB.Recordset returned will contain the fields in the entity table corresponding to the entity and entity type passed.

Signature:
Public Function getEntity(ByVal iEntityId as Integer, _
ByVal iEntityType as Integer, _
rsEntity as ADODB.Recordset) as Long

Parameters:
iEntityId: Specifies the entity requested.
iEntitytype: Specifies the type of entity requested.
rsEntity: An ADO Recordset which will contain the referenced entity.

The following columns are returned: all fields from the specified entityType table.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

updateEntity

Description:
This method is used to insert or update an entity record.

Signature:
Public Function updateEntity(Byval iEntityId as Integer, _
ByVal iEntityType as Integer, _
Optional sFName as String, -
Optional sLName as String, _
Optional sOrgName as String, -
Optional sDOB as String, _
Optional sSSN_FEI as String, _
Optional sUPIN_Lic as String, _
Optional iDoctor as Integer) as Long

Parameters:
iEntityId: Specifies the entity whose record is being updated. If iEntityId = 0 the it specifies that the entity is to be inserted.
iEntityType: Specifies the entity type of the entity whose record is being updated/inserted.
sFName: Specifies the first name of the entity being inserted/updated.
sLName: Specifies the last name of the entity being inserted/updated.
sOrgName: Specifies the organization name of the entity being inserted/updated.
sDOB: Specifies the DOB of the entity being inserted/updated.
sSSN_FEI: Specifies the SSN or FEI of the entity being inserted/updated.

sUPIN_Lic: Specifies the UPIN or License of the entity being inserted/updated. License for pharmacy, UPIN for doctor, license for Pharmacy.
iDoctor: Specifies the parent doctor when inserting a doctor. Specifies the doctor for a patient when inserting a doctor record.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getPatientInsurance

Description:
The two dimensional array or ADODB.Recordset returned will contain the insurance information for the specified patient. Multiple records may be returned where each record

Signature:
Public Function get Patient Insurance (ByVal iPatientId as Integer, _
rsInsurance as ADODB.Recordset) as Long

Parameters:
iPatientId: Specifies the patient whose insurance is requested.
rsInsurance: An ADO Recordset which will contain the patient's insurance information. Each row will contain an insurance record for the patient.

The following columns are returned: insurance_id, policy, name, FEI

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

updatePatientInsurance

Description:
Adds or updates an Insurance record for an existing patient.

Signature:
Public Function updatePatientInsurance(ByVal iPatientId as Integer, _
ByVal iInsurance Id as Integer, -
Optional sPolicy as String, _
Optional sName as String, _
Optional sFEI as String) as Long

Parameters:
iPatientId: Specifies the patient for whom the insurance information is requested.
iInsuranceId: Specifies the Insurance company inserted/updated for the patient. If the value for iInsurance_Id = 0, then this method will insert a new insurance company record.
sPolicy: Specifies the policy number for the insurance company and patient.
sName: Specifies the name of the Insurance company that is being inserted/updated
sSFEI: Specifies the FEI for the Insurance company being inserted/updated.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getAddressInfo

Description:
The two dimensional array. or ADODB.Recordset returned will contain the address information for the entity being passed. Each record will represent one address type for the entity. An entity may have 0 to many address records, but each one will have a different address type.

Signature:
Public Function getAddressInfo(ByVal iEntityId as Integer, _
ByVal iEntityTypeId as Integer, _
rsAddressInfo as ADODB.Recordset) as Long

Parameters:
iEntityId: Specifies the entity whose address is requested.
iEntityTypeId: Specifies the entitytype of the entity whose address is requested.
rsAddressinfo: An ADO Recordset which will contain the referenced entities address records.

The following columns are returned: name, addressType_id, address1, address2, city, stateCode, zip, country.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

updateAddressinfo

Description:
This method will insert or update an address record for the specified entity. If the address type parameter passed is not equal to the address type field in the address table for that entity and entity type, then a new address record will be created. If not, the address record identified by the three passed fields will be updated.

Signature:
Public Function updateAddressInfo(
ByVal iEntityId as Integer, _
ByVal iEntityTypeId as Integer, _
ByVal iAddressTypeId as Integer, _
Optional sAddress1 as String, _
Optional sAddress2 as String, _
Optional sCity as String,
Optional sStateCode as String, _
Optional sZip as String, _
Optional sCountry as String) as Long

Parameters:

iEntityId: Specifies the entity id of the entity whose address is being inserted/updated.
iEntityTypeId: Specifies the type of entity whose address is being inserted/updated.
iAddressTypeId: Specifies the address type of the entity whose address is being inserted/updated. If this value is not equal to an existing address record, then the address record is an insert. If not it is an update.
sAddress1 : Specifies the first line of the address of the entity whose address is being inserted/updated.
sAddress2: Specifies the second line of the address of the entity whose address is being inserted/updated.
sCity: Specifies the city of the entity whose address is being inserted/updated.
sStateCode: Specifies the state of the entity whose address is being inserted/updated.
sZip: Specifies the postal code of the entity whose address is being inserted/updated.
sCountry: Specifies the country of the entity whose address is being inserted/updated.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getContactInfo

Description:
The two dimensional array or ADODB.Recordset returned will contain the contact information for the entity being passed. Each record will represent one contact type for the entity. An entity may have 0 to many contact records, but each one will have a different contact type.

Signature:
```
Public Function getContactInfo(ByVal iEntityId as Integer, _
                               ByVal iEntityTypeId as Integer, _
                               rsContactInfo as ADODB.Recordset) as Long
```

Parameters:
iEntityId: Specifies the entity whose contact info is requested.
iEntityTypeId: Species the entitytype of the entity whose contact info is requested.
rsContactInfo: An ADO Recordset which will contain the referenced entities contact information.

The following columns are returned: contact_type, value.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

updateContactInfo

Description:
This method will insert or update a contact record for the specified entity. If the contact type parameter passed is not equal to the contact type field in the contact table for that entity and entity type, then a new contact record will be created. If not, the contact record identified by the three passed fields will be updated.

Signature:
Public Function updateContactInfo(
ByVal iEntityId as Integer, _
ByVal iEntityTypeId as Integer, _
ByVal iContactTypeId as Integer, -
Optional sValue as String) as Long

Parameters:
iEntityId: Specifies the entity id of the entity whose address is being inserted/updated.
iEnftyTypeId: Specifies the type of entity whose address is being inserted/updated.
iContactTypeId: Specifies the address type of the entity whose address is being inserted/updated. If this value is 0 then the address record is an insert. If not it is an update.
sValue: Specifies the value of the contact type for the entity whose contact is being inserted/updated.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

bill

Description:
This method is called after a patient's record is accessed, where the patient's doctor_id is not equal to the doctor_id of the current user, or the current user is not a doctor. This method is called after a progress note record is accessed where the progress note's doctor_id is not equal to the doctor_id of the current user, or the current user is not a doctor.

Signature:
Public Function bill(ByVal iEntityId as Integer, _ _
ByVal iEntityType as Integer, _
ByVal iTransactionType as Integer) as Long

Parameters:
iEntityId: Specifies an array of entity id of the entities being billed.
iEntityTypeId: Specifies an array of entityType ids of the entity being billed.
iTransactionType: Specifies the transaction type; patient access, note access, pharmaceutical report.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

get Bills

Description:
This method will first get all bill records that are over 30 days old and do not have a receipt record. An AR transaction of type, aged receivables, will be created in the amount of 10% of the bill amount. The bill will then be updated with the new amount, based on the original bill

amount plus the new transaction amount. This method will then get all AR records that do not have a bill (batch) number, group them by entityId and entityType, and sum them on amount. These groups will then be inserted as new bill records. Each of these bill records, including all newly created and all updated bill will be returned. If the bDetail flag is true, all corresponding AR records that make up each bill will also be returned

Signature:
Public Function getBills(rsBills as ADODB.Recordset
bDetail as Boolean) as Long

Parameters:
rsBills: Contains a list of Bill records, grouped by entity_ID and entity-type

The following columns are returned: EntityId, EntitytypeID, EntityFName, EntityLName, EntityOrgName, Bill_Date, Amount, Tran_Date, Tran_Type, Acct_Rec.Amount.

bDetail: Specifies whether the returned recordset will contain A/R records or just bill (summary) records.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getAcctsPayable

Description:
This method will return all AP records that do not have a check (batch) number.

Signature:
Public Function getAcctsPayable(rsAP as ADODB.Recordset) as Long

Parameters:
rsAP: Contains a list of AP records, grouped by entity_ID and entity_type

The following columns are returned: EntityId, EnftytypeID, EntityFName, EnftyLName, EntityOrgName, Tran_Date, Tran_Amount.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

processChecks

Description:
This method will create a check record and update the AP record with the batch number of the created check.

Signature:
Public Function processChecks(iEntityId() as Integer, _

iEntityTypeId() as Integer, _
vCheckDate() as Variant, _
iCheckNumber() as Integer, _
vCheckAmount() as Variant) as Long

Parameters:
iEntityId(): Specifies an array of entity id of the entities whose check records are being updated.
iEntityTypeId(): Specifies an array of entityType ids of the entities whose check records are being updated.
vCheckDate: Specifies an array of the dates of the check records being updated.
ICheckNumber: Specifies an array of the check numbers of the check records being updated.
vCheckAmount: Specifies an array of the amounts of the check records being updated.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

processReceipts

Description:
This method will update the Payment Received table with payments amounts and dates for the corresponding bills.

Signature:
Public Function processReceipts(iEntityId() as Integer, _
iEntityTypeId() as Integer, _
vRecDate() as Variant, _
vRecAmount() as Variant) as Long

Parameters:
iEntityIdo: Specifies an array of entity id of the entities whose bills are being paid.
iEntityTypeId(): Specifies an array of entityType ids of the entities whose bills are being paid.
vRecDate: Specifies an array of the dates of the receipts for the bills that were paid.
vRecAmount: Specifies an array of the amounts of the receipts for the bills that were paid.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getReceipts

Description:
This method will return all Payment Received records where there is no Deposit id.

Signature:
Public Function getReceipts(rsReceipts as ADODB.Recordset) as Long

Parameters:
rsReceipts: Contains a list of Payment Received records.

The following columns are returned: EntityId, EntitytypeID, EntityFName, EnftyLName, EnftyOrgName, ReceiptID, Receipt_Date,, Amount.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

updateDeposit

Description:
This method will create a deposit record and update the receipt records included on that deposit with the deposit id.

Signature:
Public Function updateDeposit(iReceiptId() as Integer, _
sDepDate as String, _
sDepTicketId as String) as Long

Parameters:
iReceiptId: Specifies an array of Receipts to be updated with the deposit id.
sDepDate: Specifies the date of the deposit.
sDepTicketId: Specifies the deposit ticket id for the deposit.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

getReconciliation

Description:
This method will return all deposits records with no clear_date and all checks with no check_clear_date.

Signature:
Public Function getReconciliation(rsChecks as ADODB.Recordset, _
rsDeposits as ADODB.Recordset) as Long

Parameters:
rsChecks: Contains check records where there is no check-clear-date for that check.

The following columns are returned: Entity_Id, Entity_Type, Check_Date, Check_Number, Check_Amount.

rsDeposits: Contains deposit records where there is no clear date for that deposit.

The following columns are returned: Entity_Id, Entity_Type, Deposit_Date, Deposit_Amount.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

updateReconciliation

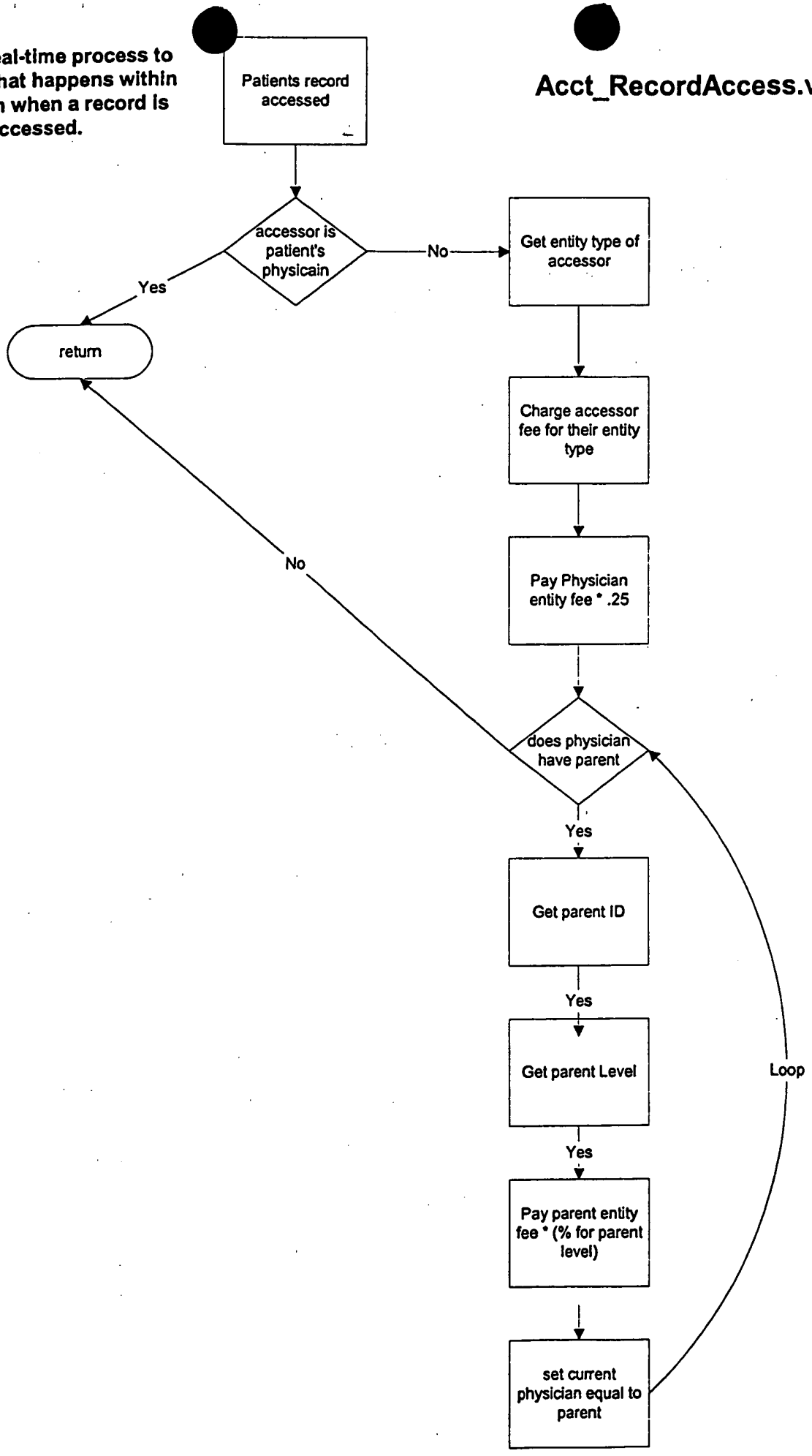
Description:
This method will update the checks table with a check_clear_date for checks that clear and updates the deposit table with a clear_date for deposits that clear.

Signature:
Public Function updateReconciliation(iBatch() as Integer, _
vCheckAmount() as Variant, _
sCheckDate() as String, _
iDepositID() as Integer, _
vDepositAmount() as Variant, _
sDepositDate() as String) as Long

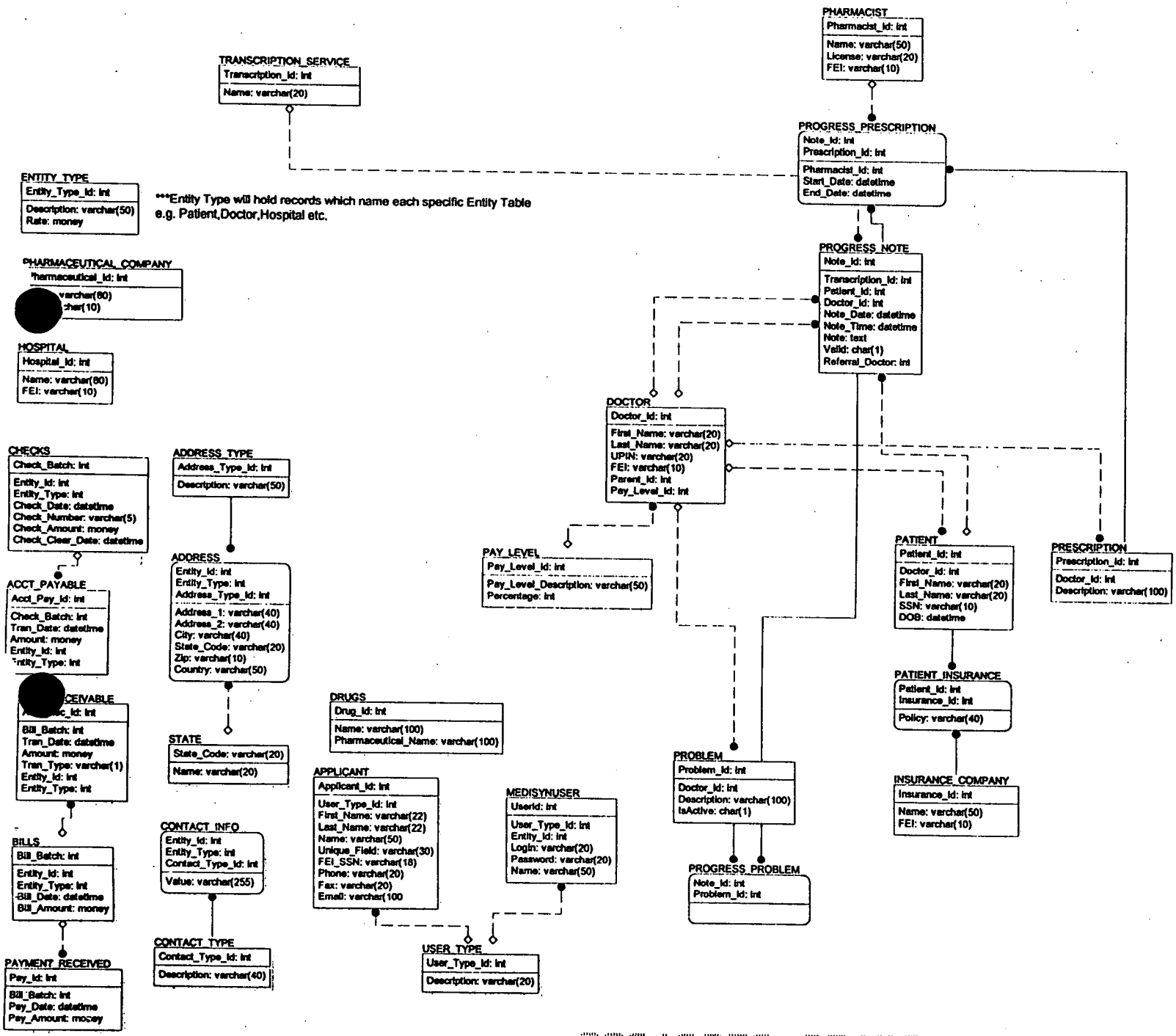
Parameters:
iBatch(): Array of check batch numbers. Each item in the array corresponds to a check amount and dates in the vCheckAmount array and the sCheckDate array.
vCheckAmount(): Specifies an array of check amounts being reconciled.
sCheckClearDate(): Specifies an array of the dates those checks cleared on the bank statement.
iDepositId(): Specifies an array of deposit ids being reconciled.
vDepositAmount(): Specifies an array of the deposit Amounts being reconciled.
sDepositClearDate(): Specifies an array of the dates those deposits cleared on the bank statement.

Return type:
Long: returns 0 if successful and an Error Code if unsuccessful.

This is a real-time process to describe what happens within the system when a record is accessed.



MEDISYN DATA MODEL



Data Dictionary

Entity Name	Entity Attribute Name	Entity Attribute Definition	Entity Attribute Column Datatype
ACCT_PAYABLE	Acct_Pay_Id	Unique sequential number referencing each record in the table	int
	Check_Batch	The specific check that this transaction is associated with. Must be valid in the CHECKS table.	
	Tran_Date	Date of account payable transaction	datetime
	Amount	Amount of account payable transaction	money
	Entity_Id	Identifies the unique record in associated EntityTable. Must be valid in associated entity table.	int
	Entity_Type	Identifies type of entity. Must be valid in ENTITYTYPE table.	
ACCT_RECEIVABLE	Acct_Rec_Id	Unique sequential number referencing each record in the table	
	Bill_Batch	The specific bill that this transaction is associated with. Must be valid in the BILLS table.	
	Tran_Date	Date of account receivable transaction	datetime
	Amount	amount of account payable transaction	money
	Tran_Type	Type of account receivable transaction. Valid values are: O - Overpaid, U - Underpaid, R - Record Access	varchar(1)
	Entity_Id	Identifies the unique record in associated EntityTable. Must be valid in associated entity table.	int
	Entity_Type	Identifies type of entity. Must be valid in ENTITYTYPE table.	
ADDRESS	City	The applicable city.	varchar(40)
	Address_2	The second line of the street address.	
	State_Code	Code identifyingstate Must be valid in STATE table.	varchar(20)
	Zip	The standard postal zip code.	varchar(10)
	Country	Name of Country.	varchar(50)
	Entity_Id	Identifies the unique record in associated EntityTable. Must be valid in associated entity table.	int
	Address_1	The first line of the street address.	varchar(40)
	Entity_Type	Identifies type of entity. Must be valid in ENTITYTYPE table.	int
	Address_Type_Id	Code identifying type of Address. Must be valid in ADDRESS_TYPE table.	
ADDRESS_TYPE		Unique sequential number referencing each record in the table	
	Description	The textual description identifying a specific type of address.	varchar(50)
BILLS	Bill_Batch	Unique sequential number referencing each record in the table	int
	Entity_Id	Identifies the unique record in associated EntityTable. Must be valid in associated entity table.	
	Entity_Type	Identifies type of entity. Must be valid in ENTITYTYPE table.	
	Bill_Date	Date Bill was created.	datetime
	Bill_Amount	Amount of Bill.	money
CHECKS	Check_Batch	Unique sequential number referencing each record in the table	int
	Entity_Id	Identifies the unique record in associated EntityTable. Must be valid in associated entity table.	
	Entity_Type	Identifies type of entity. Must be valid in ENTITYTYPE table.	
	Check_Date	Date Check was created	datetime
	Check_Number	Number of Check	varchar(5)

09616276 .07140D

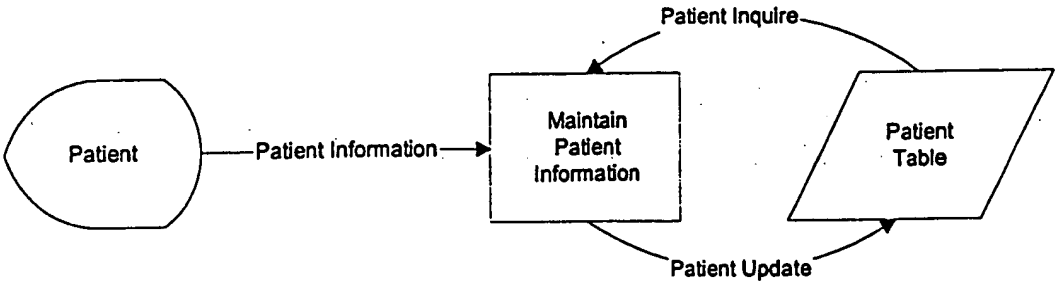
Entity Name	Entity Attribute Name	Entity Attribute Definition	Entity Attribute Column Datatype
CHECKS	Check_Amount	Amount of Check	money
	Check_Clear_Date	Date check cleared bank.	datetime
CONTACT_INFO	Contact_Type_Id	Code identifying type of Address. Must be valid in CONTACT_TYPE table.	int
	Value	Text field containing contact information	varchar(255)
	Entity_Type	Identifies type of entity. Must be valid in ENTITYTYPE table.	int
	Entity_Id	Identifies the unique record in associated EntityTable. Must be valid in associated entity table.	
CONTACT_TYPE	Contact_Type_Id	Unique sequential number referencing each record in the table	
	Description	The textual description identifying a specific type of contact	varchar(40)
DOCTOR	FEI	Federal Identification Number	varchar(10)
	UPIN	UPIN Number of Doctor	varchar(20)
	Parent_Id	Identifier of physician which recruited doctor. Pointer to DOCTOR Table.	int
	Pay_Level_Id	Code representing the level of pay out for the doctor. Must be valid in the PAY_LEVEL table.	
	Doctor_Id	Unique sequential number referencing each record in the table	
	Last_Name	Last Name of Doctor	varchar(20)
	First_Name	First Name of Doctor	
ENTITY_TYPE	Entity_Type_Id	Unique sequential number referencing each record in the table	int
	Description	The textual description identifying a specific type of entity.	varchar(50)
	Rate	Rate of pay for specific entity.	money
HOSPITAL	Hospital_Id	Unique sequential number referencing each record in the table	int
	Name	Name of Hospital	varchar(80)
	FEI	Federal Identification Number	varchar(10)
INSURANCE_COMPANY	Insurance_Id	Unique sequential number referencing each record in the table	int
	Name	Name of Insurance Company	varchar(50)
	FEI	Federal Identification Number	varchar(10)
MEDISYNUSER	UserId	User ID of system user	int
	Type	Type of User	varchar(10)
	Entity_Id	Identifies the unique record in associated EntityTable. Must be valid in associated entity table.	int
	Login	Login of User	varchar(20)
	Password	Password of User	
	Name	Name of System User if not in any Entity Table.	varchar(50)
PATIENT	Patient_Id	Unique sequential number referencing each record in the table	int
	First_Name	First Name of Patient	varchar(20)
	Last_Name	Last Name of Patient	
	SSN	Patients Social Security Number	varchar(10)
	DOB	Patients Date of Birth	datetime
PATIENT_INSURANCE	Patient_Id	Identifies the associated patient that this record belongs to. Must be valid in PATIENT table.	int
	Insurance_Id	Identifies the associated insurance that this record belongs to. Must be valid in INSURANCE table.	
	Policy	Number of Insurance Policy	varchar(40)
PAY_LEVEL	Pay_Level_Id	Unique sequential number referencing each record in the table	int
	Pay_Level_Description	The textual description identifying a specific pay level.	varchar(50)

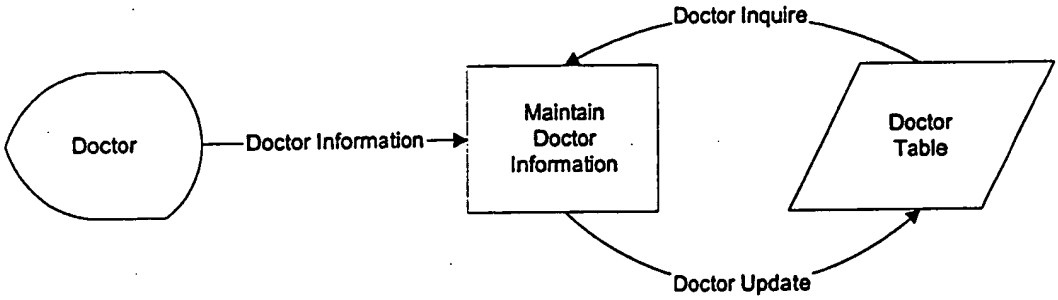
0961627 .07140

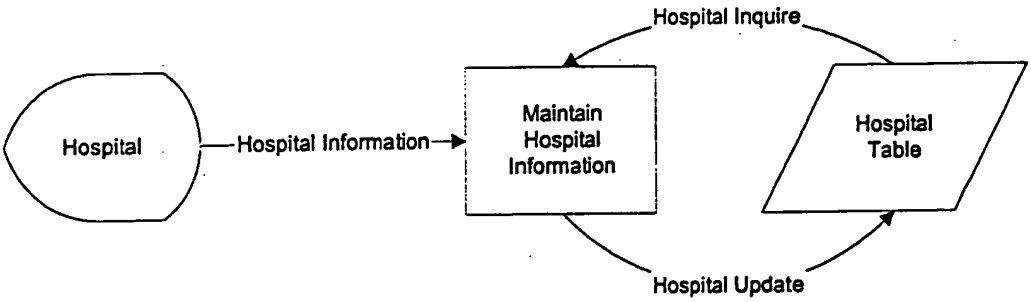
Entity Name	Entity Attribute Name	Entity Attribute Definition	Entity Attribute Column Datatype
PAY_LEVEL	Percentage	Percentage of Payout.	int
PAYMENT_RECEIVED	Pay_Date	Date of payment	datetime
	Pay_Amount	Amount of payment	money
	Bill_Batch	Identifies the associated bill that this payment belongs with.. Must be valid in BILLS table.	int
	Pay_Id	Unique sequential number referencing each record in the table	
PHARMACEUTICAL_COMPANY	Pharmaceutical_Id	Unique sequential number referencing each record in the table	
	Name	Name of Pharmaceutical Company	varchar(80)
	FEI	Federal Identification Number	varchar(10)
PHARMACIST	Pharmacist_Id	Unique sequential number referencing each record in the table	int
	Name	Name of pharmacist	varchar(50)
	License	Pharmacist License Number	varchar(20)
	FEI	Federal Identification Number	varchar(10)
PRESCRIPTION	Prescription_Id	Unique sequential number referencing each record in the table	int
	Doctor_Id	Identifies the associated doctor that this prescription belongs with. Must be valid in DOCTOR table.	
	Description	The textual description identifying a specific prescription.	varchar(100)
PROBLEM	Problem_Id	Unique sequential number referencing each record in the table	int
	Doctor_Id	Identifies the associated doctor that this prescription belongs with. Must be valid in DOCTOR table.	
	Description	The textual description identifying a specific problem.	varchar(100)
	IsActive	Flag identifying whether problem is still an active problem.	char(1)
PROGRESS_NOTE	Note_Date	Date of Progress Note	datetime
	Doctor_Id	Identifies the associated doctor for this progress note. Must be valid in DOCTOR table.	int
	Note_Time	Time of Progress Note	datetime
	Note	Textual Transcription of progress note.	text
	Valid	Flag indicating whether this progress note has been deemed valid.	char(1)
	Note_Id	Unique sequential number referencing each record in the table	int
	Patient_Id	Identifies the associated patient for this progress note. Must be valid in PATIENT table.	
	Transcription_Id	Identifies the associated transcription service that input this progress note transcription. Must be valid in TRANSCRIPTION SERVICE table.	
	Referral_Doctor_Id	Identifies the associated doctor that this patient was referred to. Must be valid in REFERRAL DOCTOR table.	
PROGRESS_PRESCRIPTION	Note_Id	Identifies the associated progress note. Must be valid in PROGRESS NOTE table.	
	Prescription_Id	Identifies the associated prescription for this progress note. Must be valid in PRESCRIPTION table.	
	Pharmacist_Id	The pharmacist filling the prescription. Must be valid in PHARMACIST table.	
	Start_Date	The start date of the prescription.	datetime
	End_Date	The end date of the prescription.	
PROGRESS_PROBLEM	Note_Id	Identifies the associated progress note. Must be valid in PROGRESS NOTE table.	int

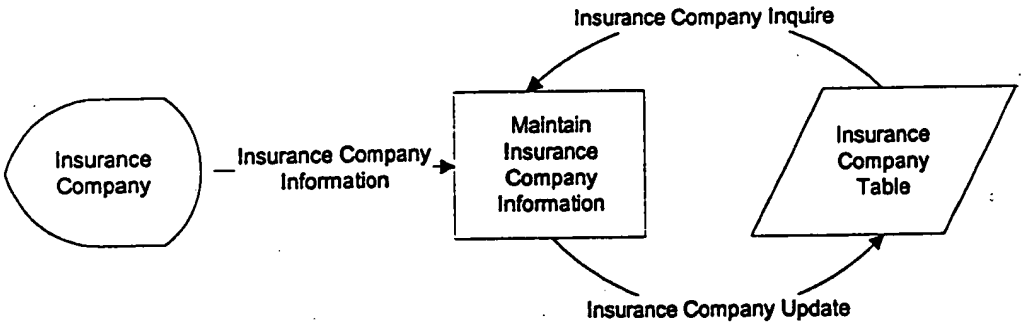
Entity Name	Entity Attribute Name	Entity Attribute Definition	Entity Attribute Column Datatype
PROGRESS_PROBLEM	Problem_Id	Identifies the associated prescription for this progress note. Must be valid in PROBLEM table.	int
REFERRAL_DOCTOR	Referral_Doctor_Id	Unique sequential number referencing each record in the table	
	First_Name	First Name of Referral Doctor	varchar(20)
	Last_Name	Last Name of Referral Doctor	
	Office_Name	Name of doctor's office	varchar(50)
	UPIN	UPIN Number	varchar(20)
STATE	State_Code	Unique code referencing each state in the table	
	Name	The textual description of a state.	
TRANSCRIPTION_SERVICE	Transcription_Id	Unique sequential number referencing each record in the table	int
	Name	Name of the Transcription Service.	varchar(20)

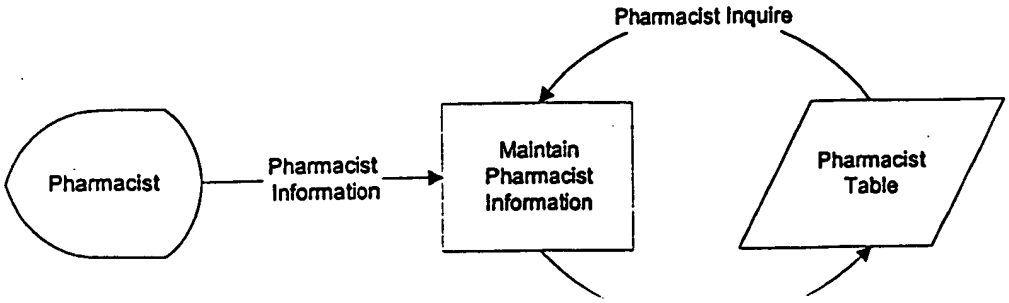
09616276 .071400

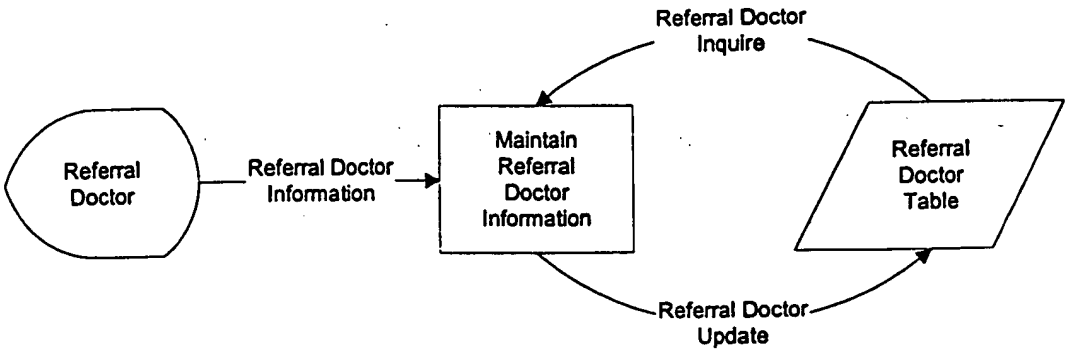

Patient Inquire
Patient
Patient Information
Maintain Patient Information
Patient Table
Patient Update

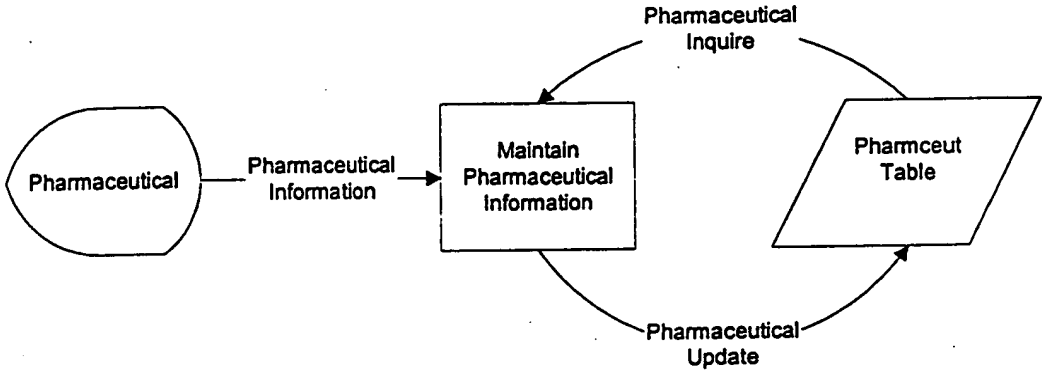

Doctor Inquire
Doctor
Doctor Information
Maintain Doctor Information
Doctor Table
Doctor Update

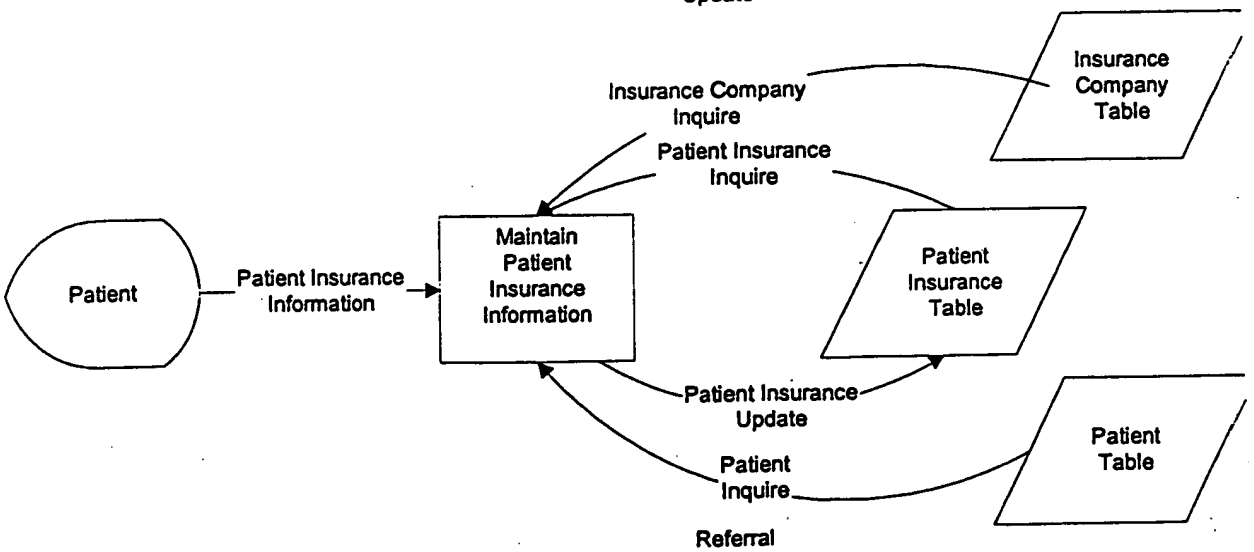

Hospital Inquire
Hospital
Hospital Information
Maintain Hospital Information
Hospital Table
Hospital Update

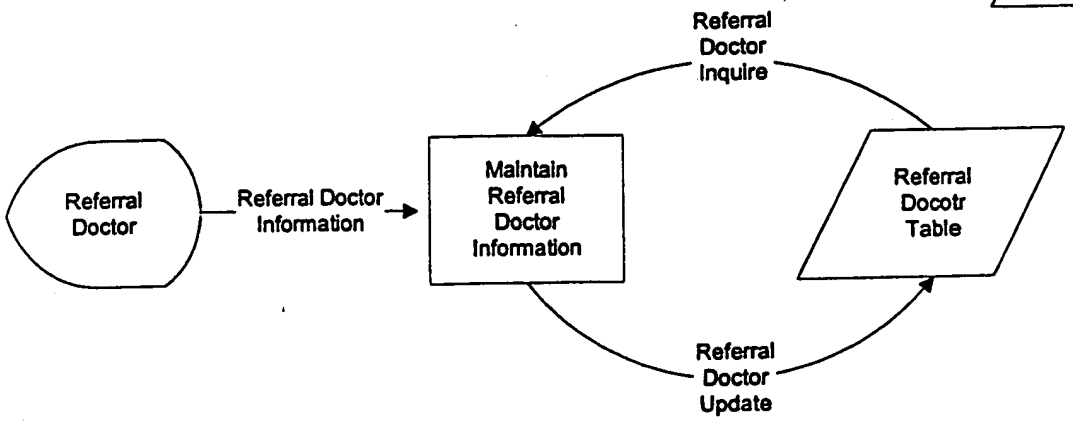

Insurance Company Inquire
Insurance Company
Insurance Company Information
Maintain Insurance Company Information
Insurance Company Table
Insurance Company Update


Pharmacist Inquire
Pharmacist
Pharmacist Information
Maintain Pharmacist Information
Pharmacist Table

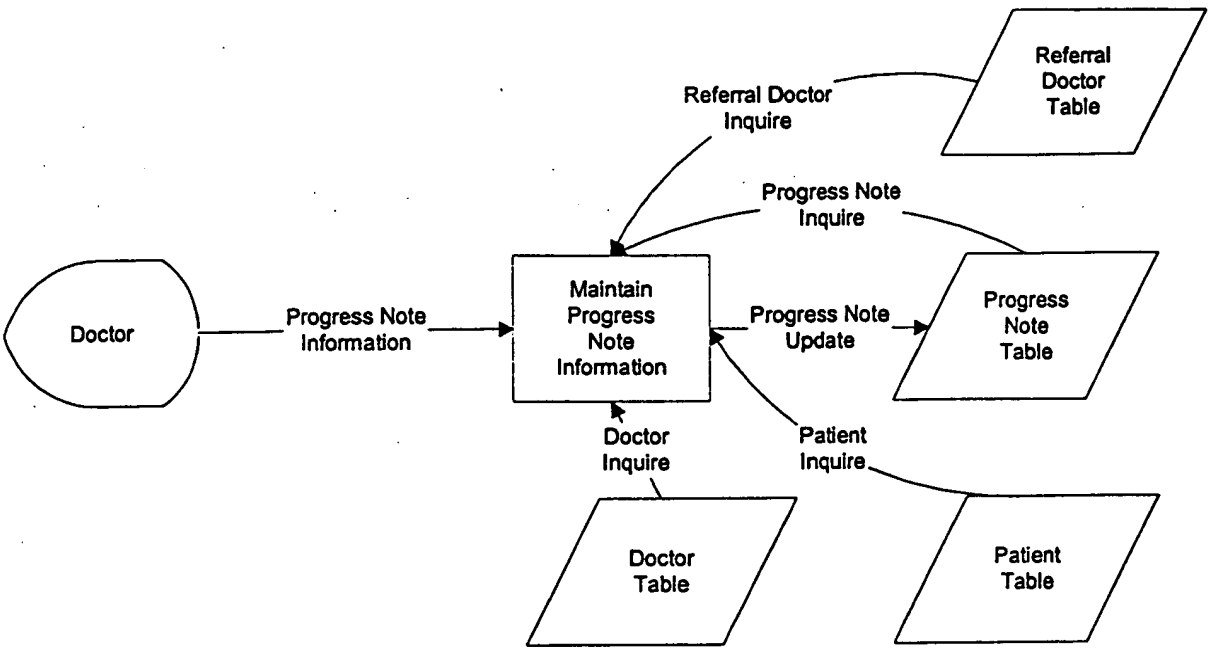


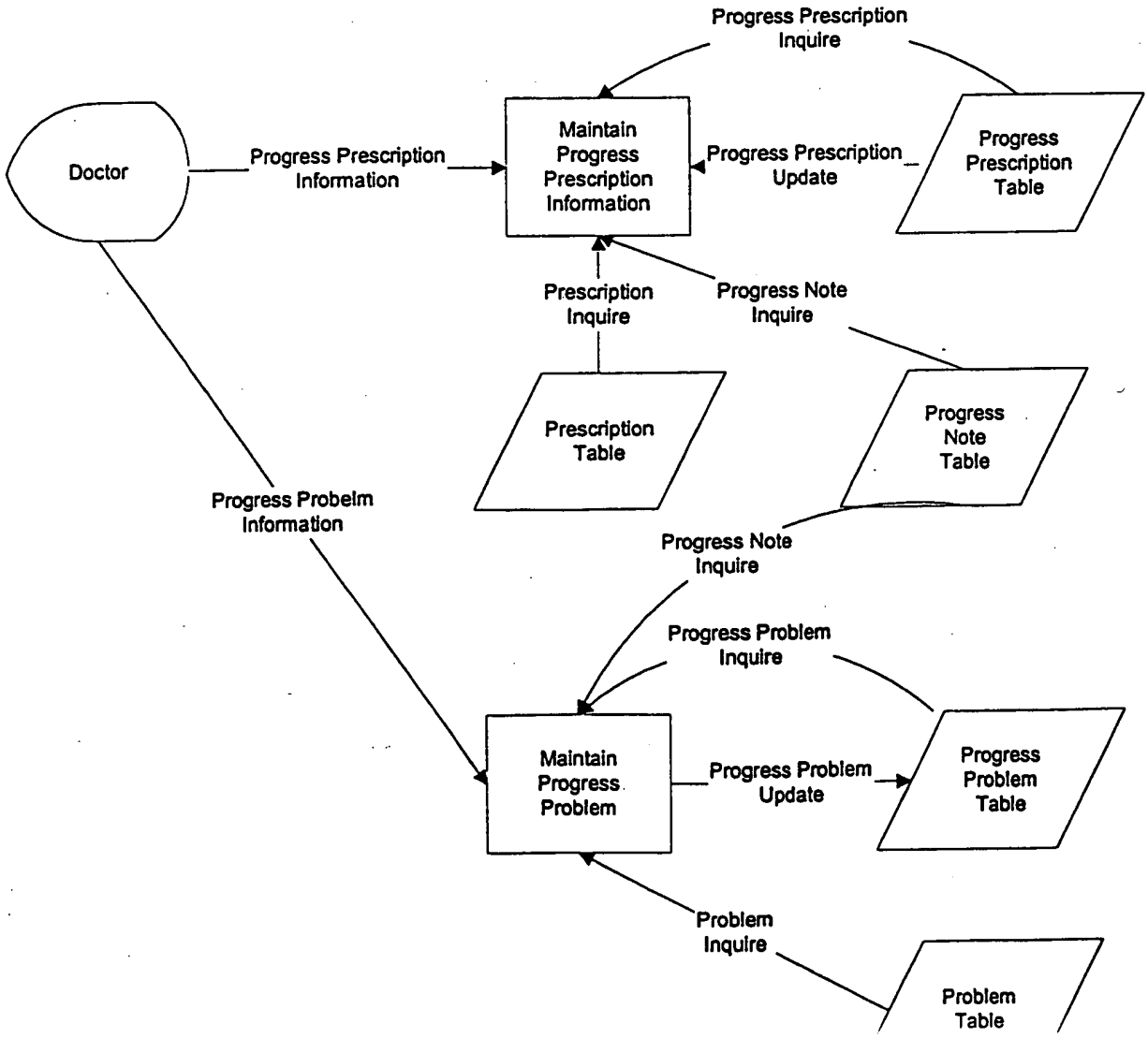


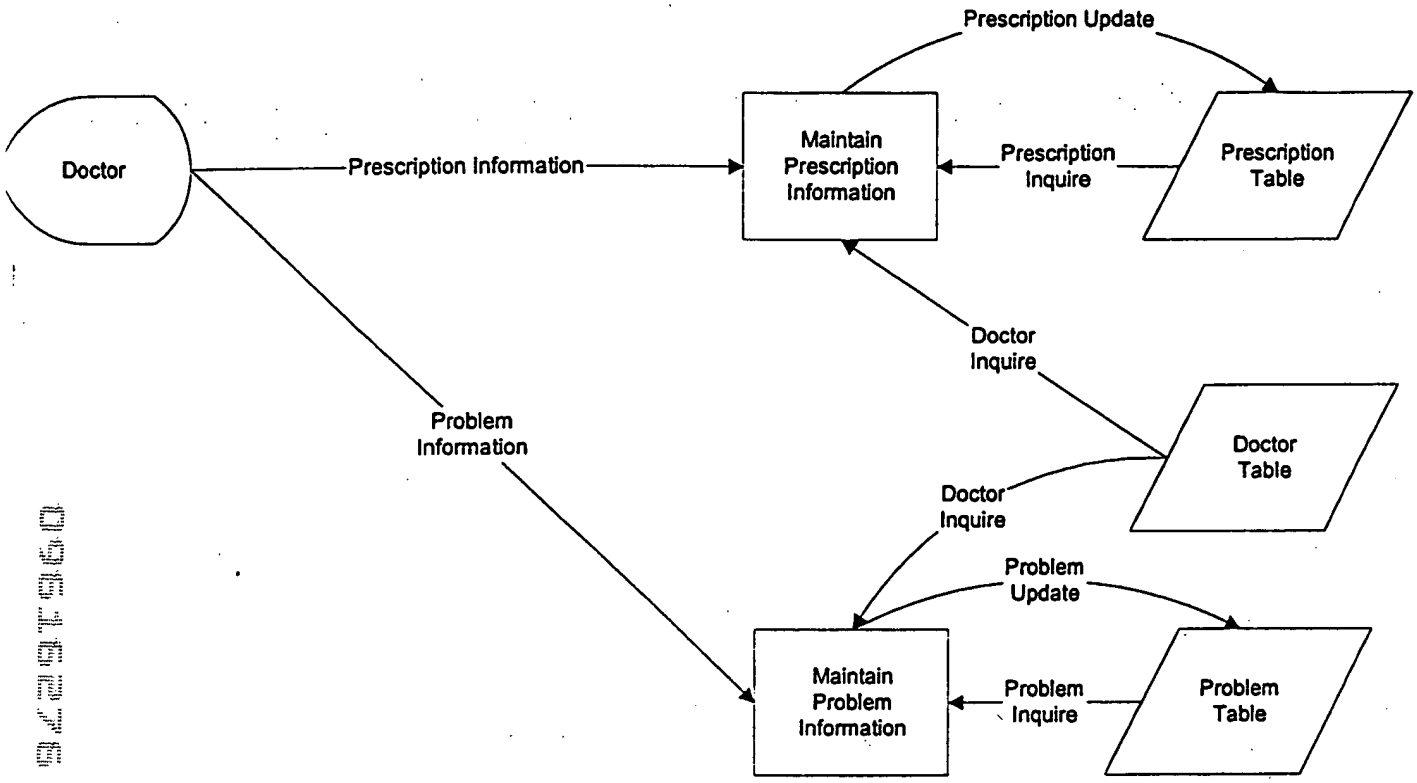




CONFIDENTIAL

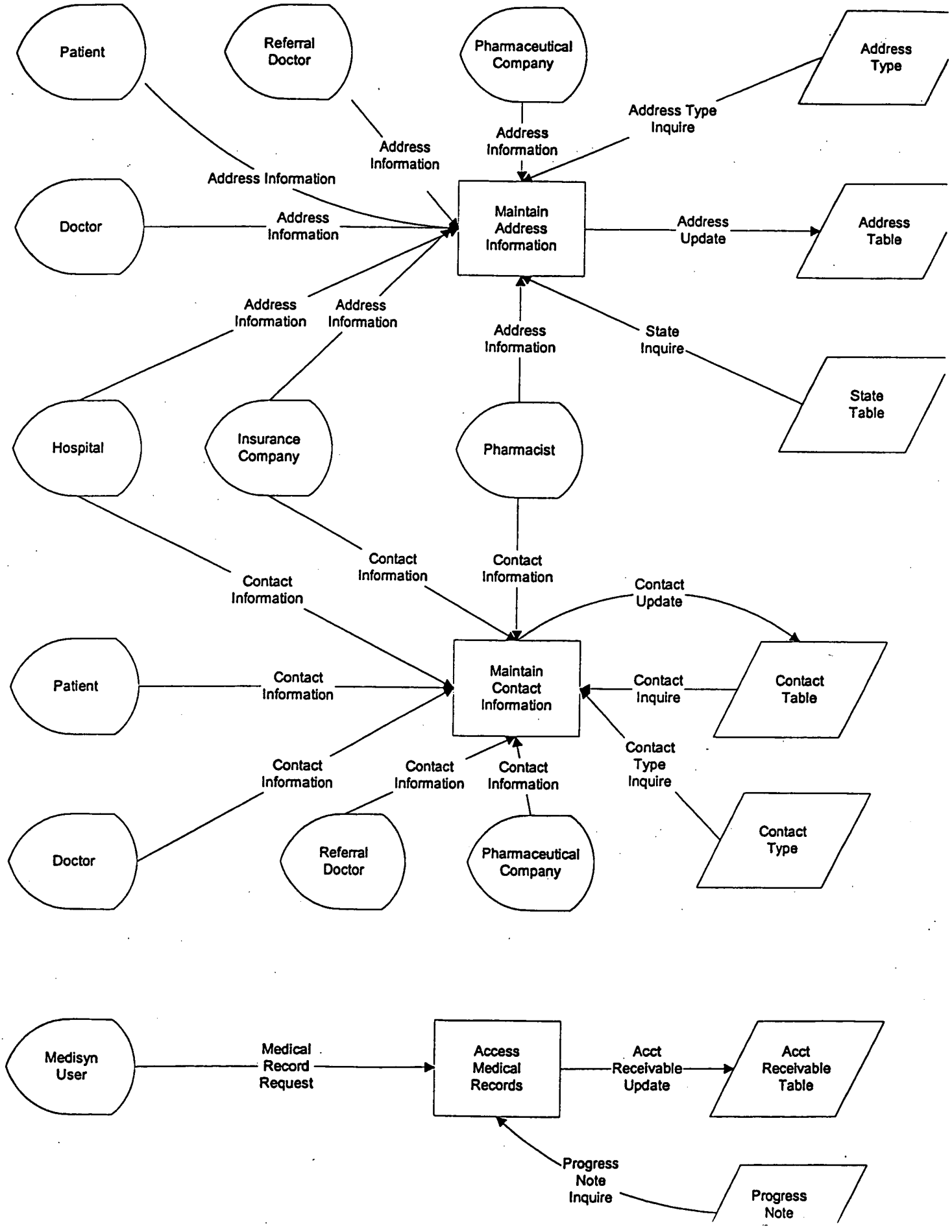

Referral Doctor Table
Referral Doctor Inquire
Progress Note Inquire
Doctor
Progress Note Information
Maintain Progress Note Information
Progress Note Update
Progress Note Table
Doctor Inquire
Patient Inquire
Doctor Table
Patient Table

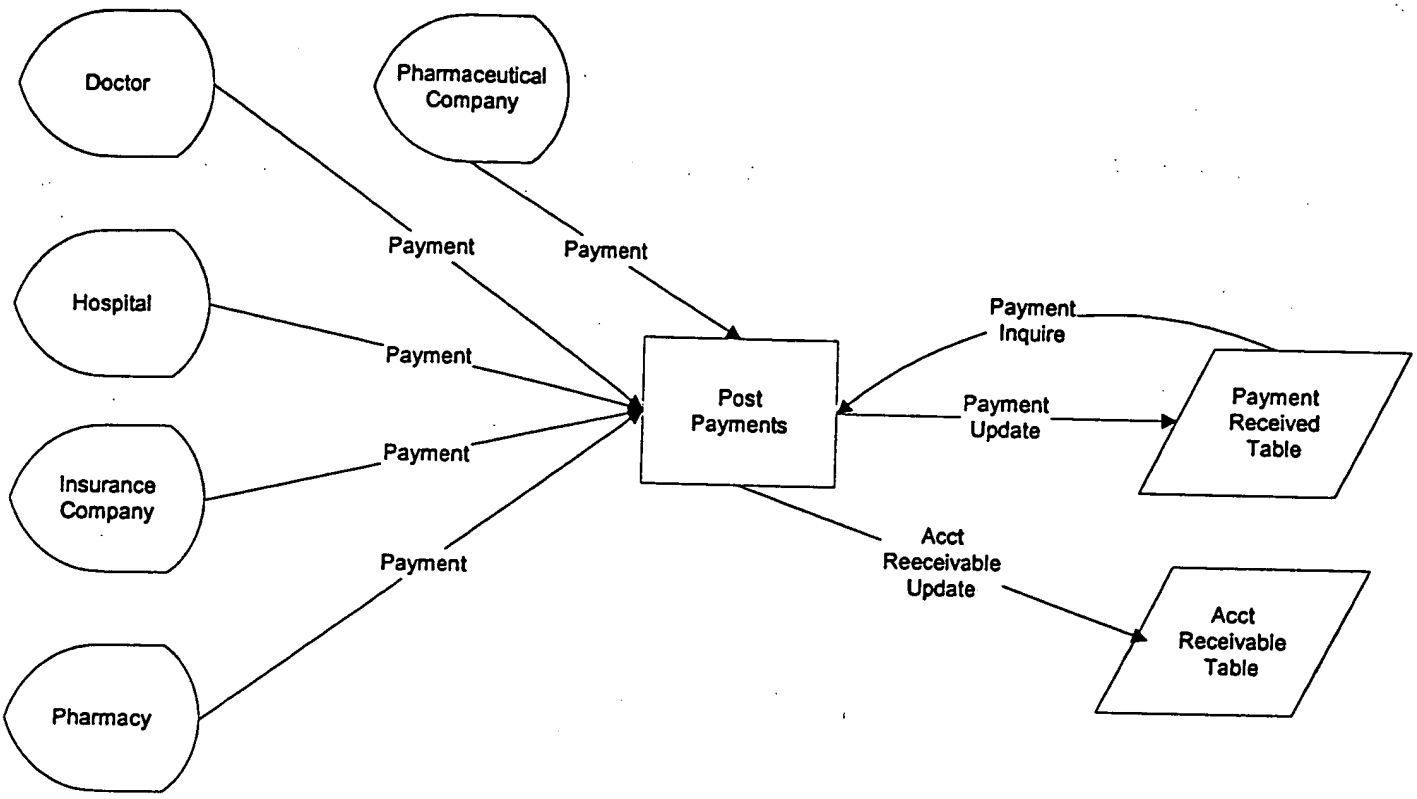

Progress Prescription Inquire
Doctor
Progress Prescription Information
Maintain Progress Prescription Information
Progress Prescription Update
Progress Prescription Table
Prescription Inquire
Progress Note Inquire
Prescription Table
Progress Note Table
Progress Probelm Information
Progress Note Inquire
Progress Problem Inquire
Maintain Progress Problem
Progress Problem Update
Progress Problem Table
Problem Inquire
Problem Table



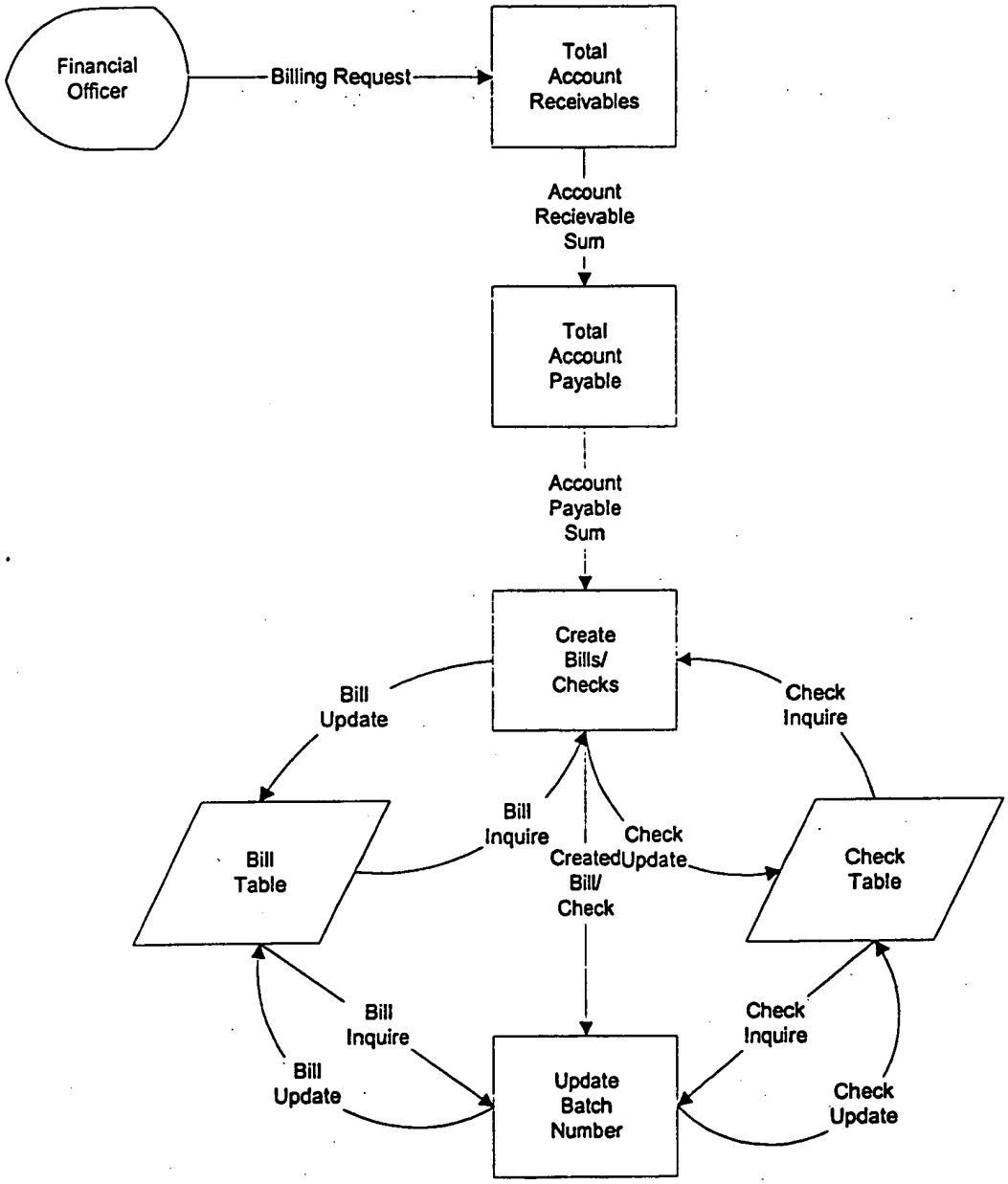
Doc Id: 5 Type: DRW Group: 1 Page: 1 of 1

MEDISYN

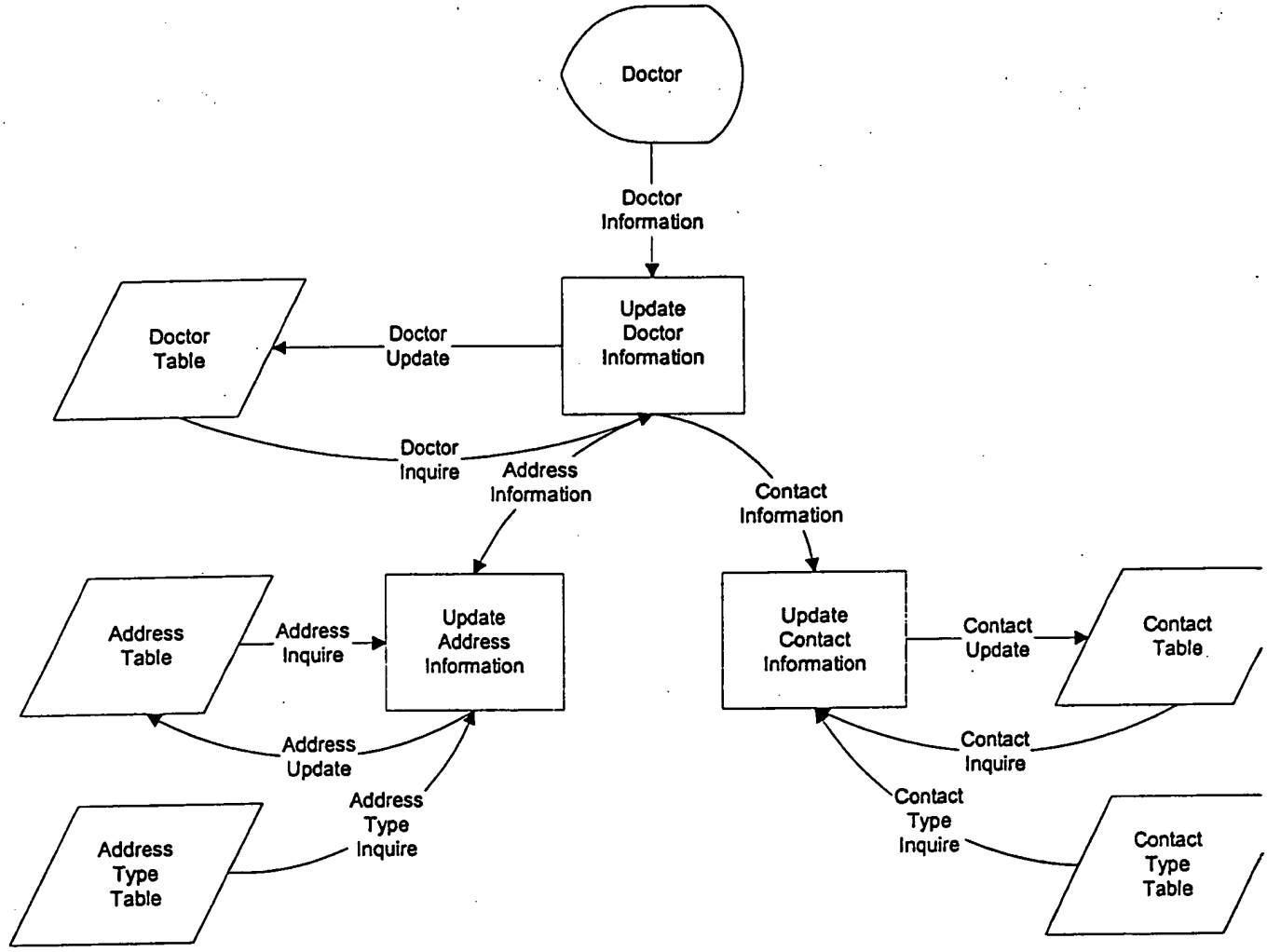

Patient
Referral Doctor
Pharmaceutical Company
Address Type
Address Type Inquire
Address Information
Address Information
Address Information
Doctor
Address Information
Maintain Address Information
Address Update
Address Table
Address Information
Address Information
Address Information
State Inquire
State Table
Hospital
Insurance Company
Pharmacist
Contact Information
Contact Information
Contact Information
Contact Information
Contact Update
Patient
Contact Information
Maintain Contact Information
Contact Inquire
Contact Table
Contact Information
Contact Information
Contact Information
Contact Type Inquire
Doctor
Referral Doctor
Pharmaceutical Company
Contact Type
Medisyn User
Medical Record Request
Access Medical Records
Acct Receivable Update
Acct Receivable Table
Progress Note Inquire
Progress Note

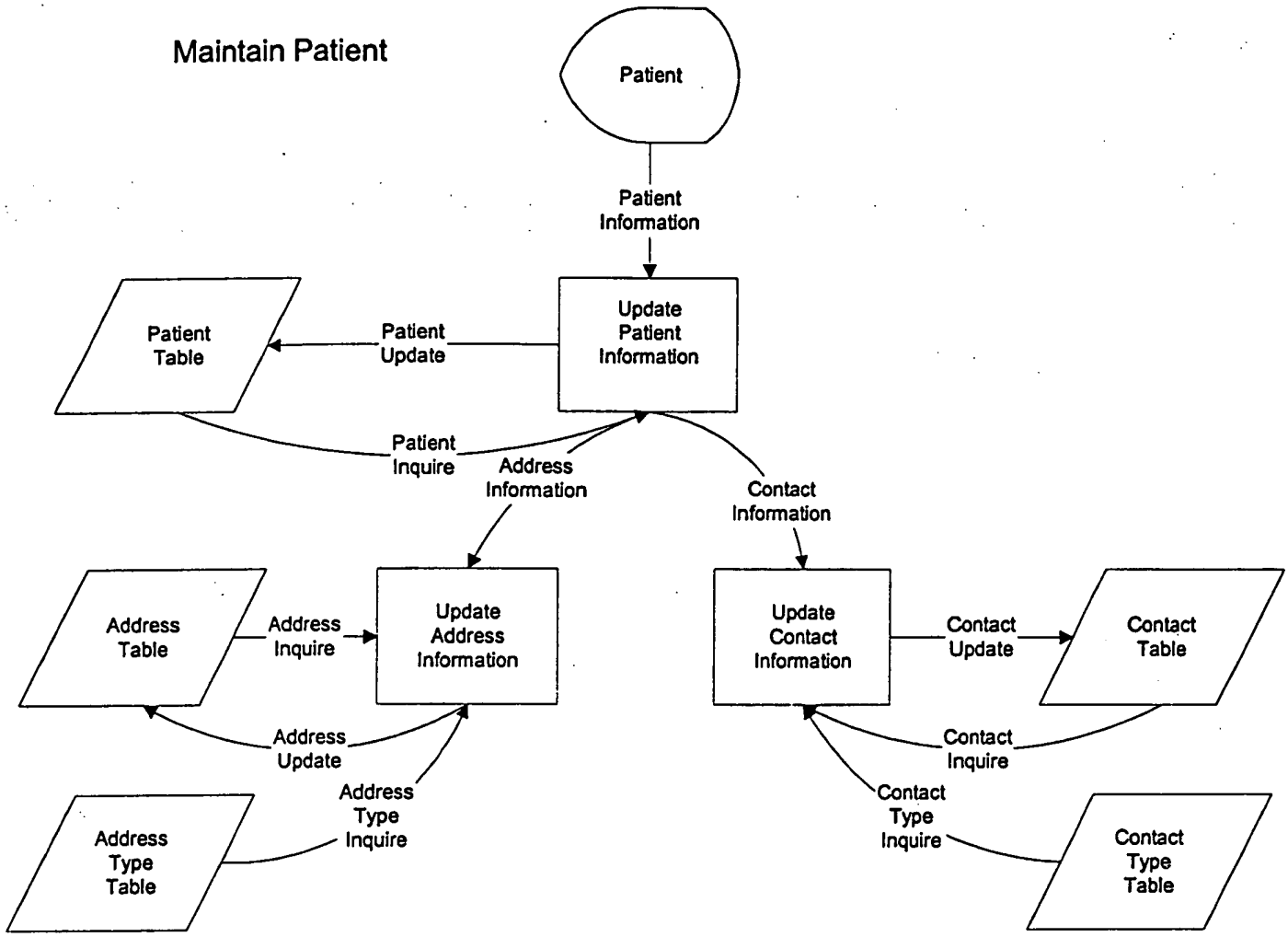

Doctor
Pharmaceutical Company
Hospital
Insurance Company
Pharmacy
Payment
Payment
Payment
Payment
Payment
Post Payments
Payment Inquire
Payment Update
Payment Received Table
Acct Reeceivable Update
Acct Receivable Table

0050429


Financial Officer
Billing Request
Total Account Receivables
Account Recievable Sum
Total Account Payable
Account Payable Sum
Create Bills/ Checks
Bill Update
Check Inquire
Bill Table
Bill Inquire
Check Update
Created Bill/ Check
Check Table
Bill Inquire
Bill Update
Check Inquire
Check Update
Update Batch Number

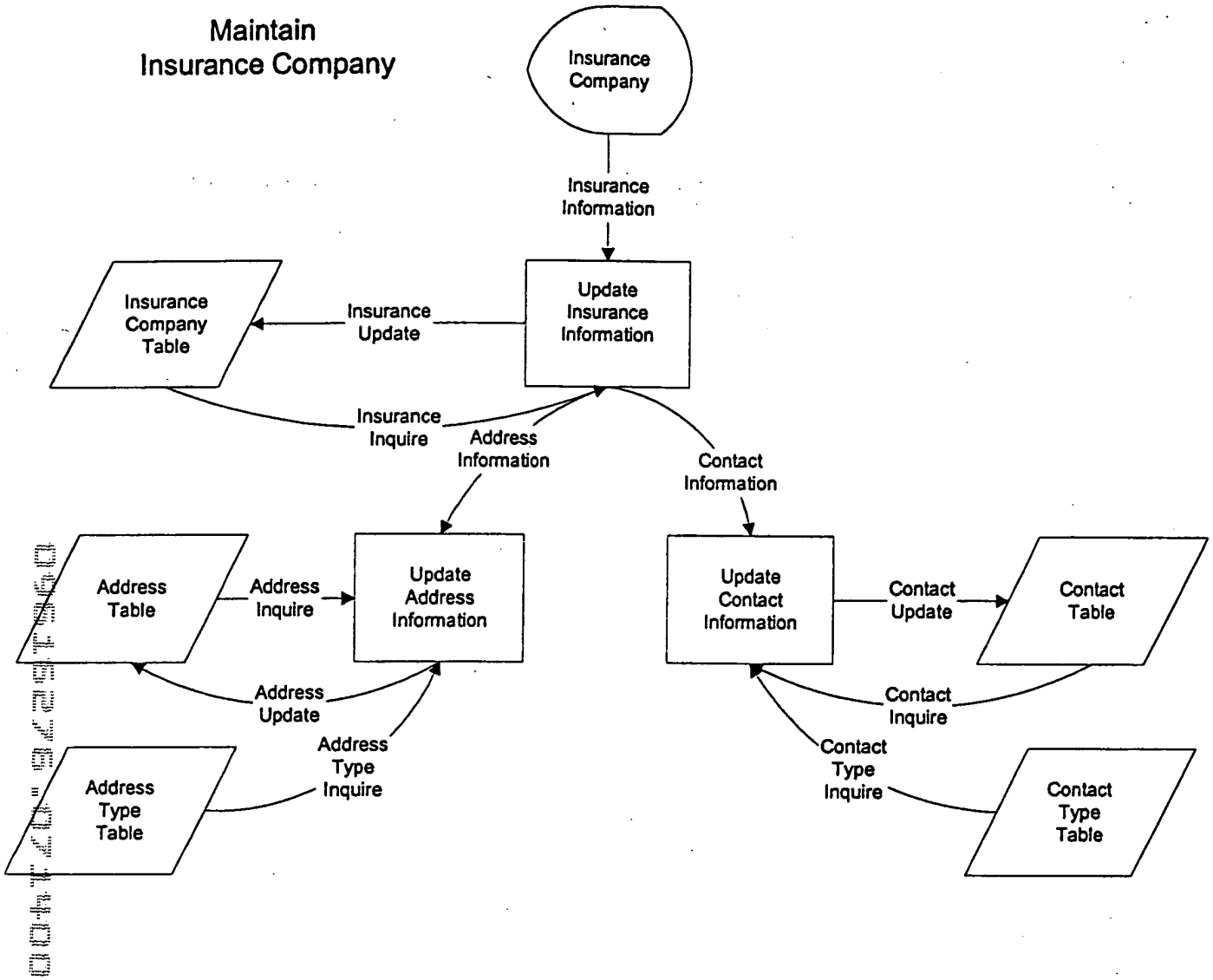
Maintain Doctor

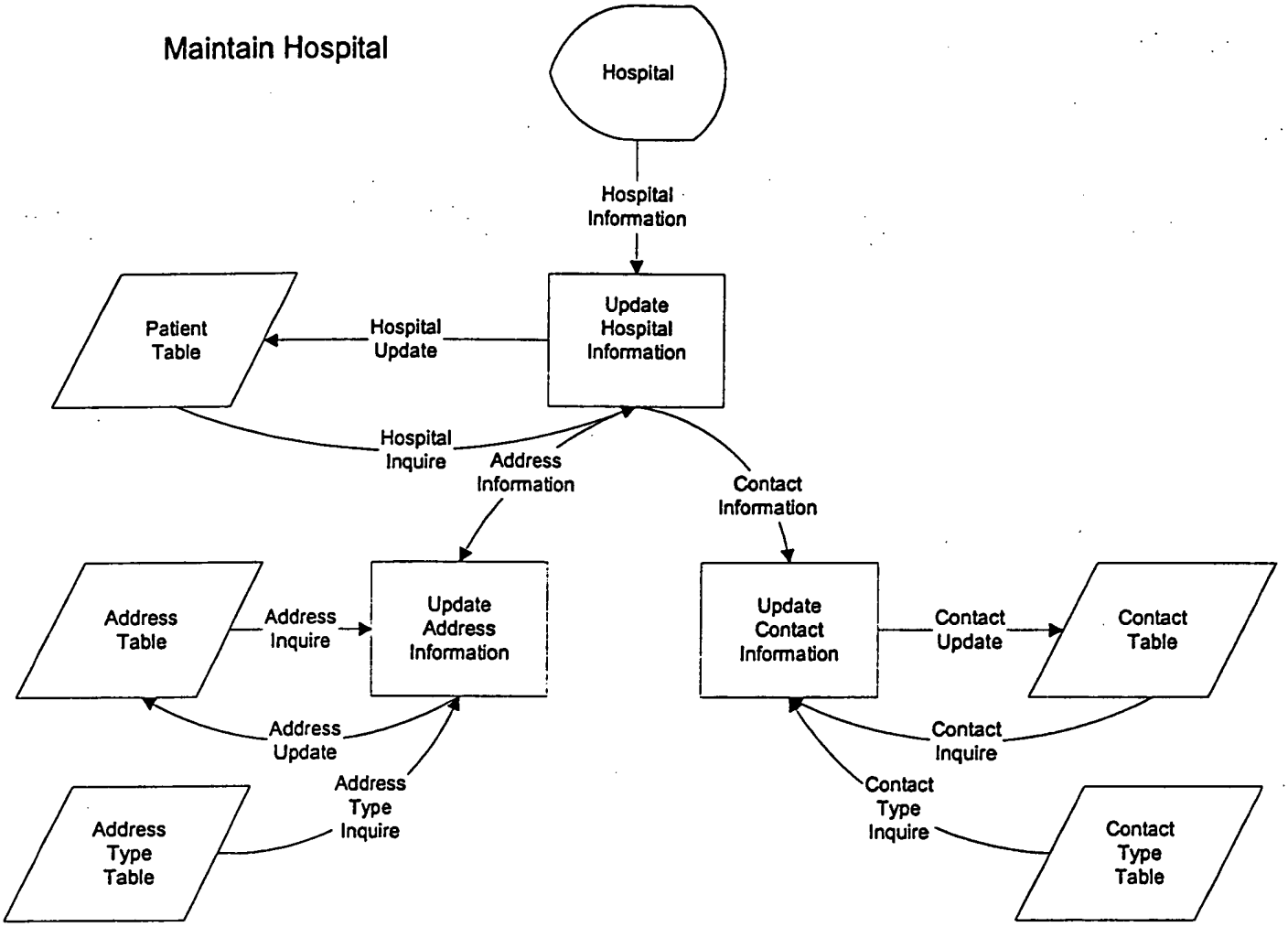




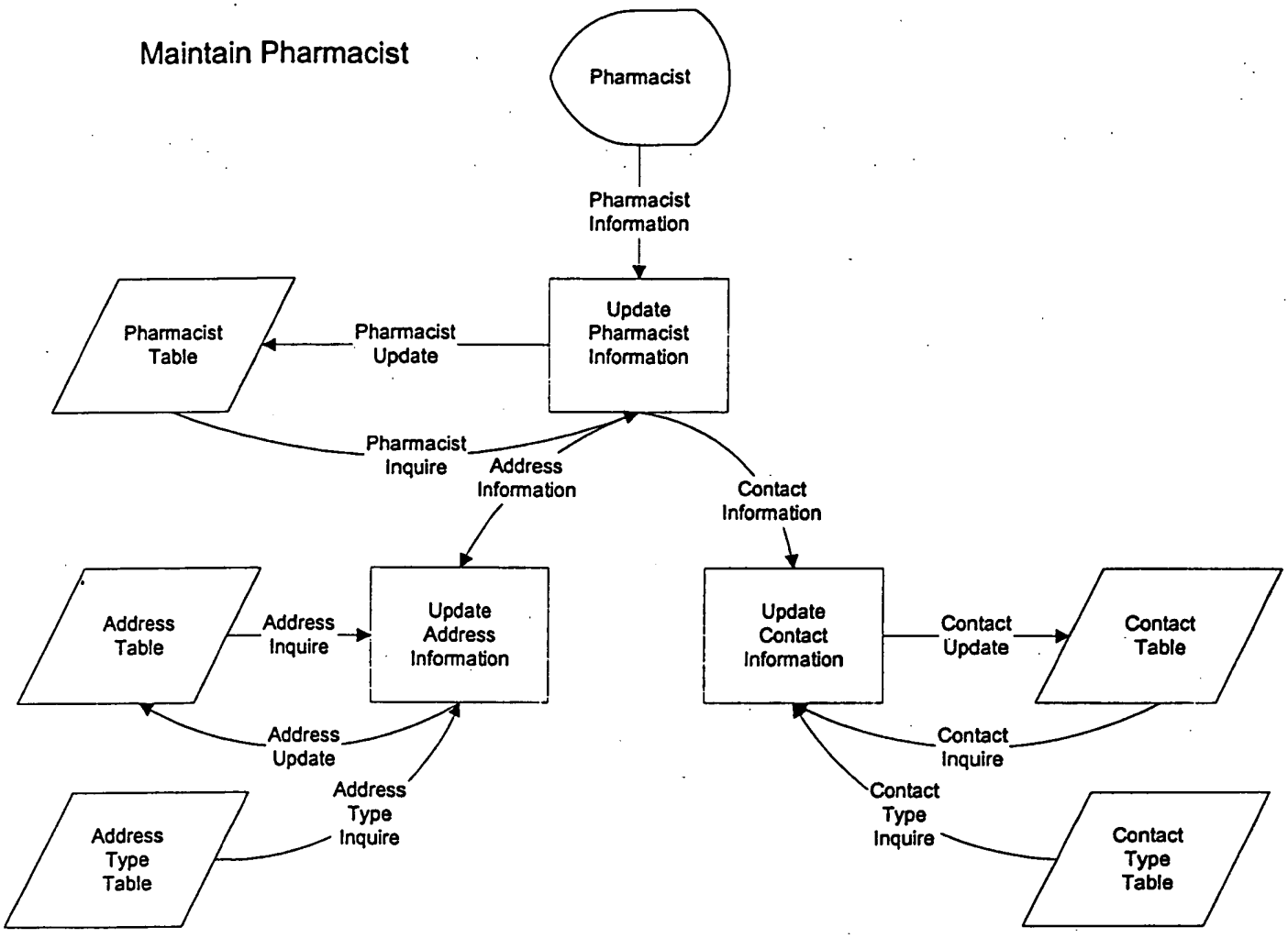
Confidential - Property of Medisyn

Maintain Insurance Company

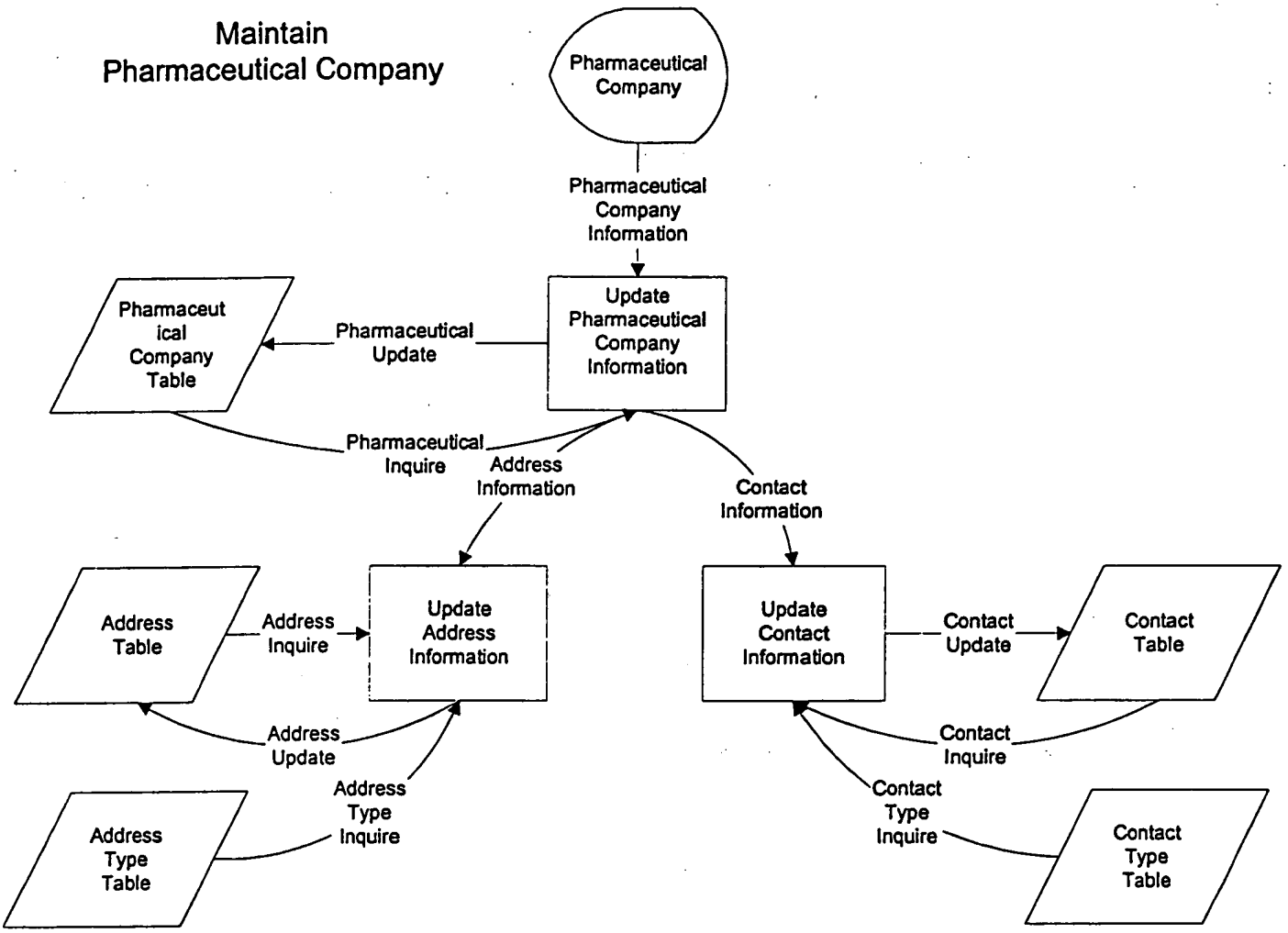




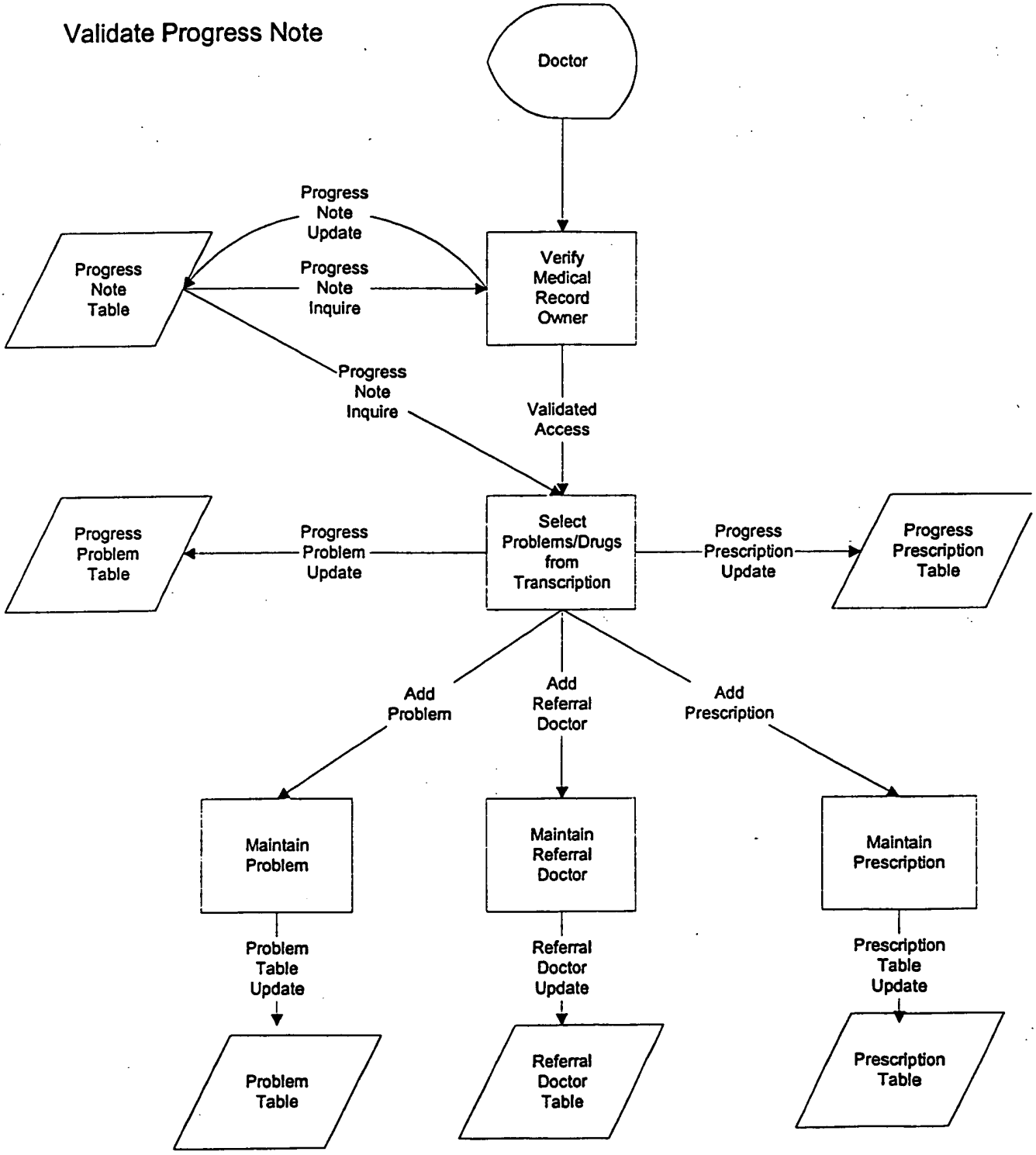
Confidential



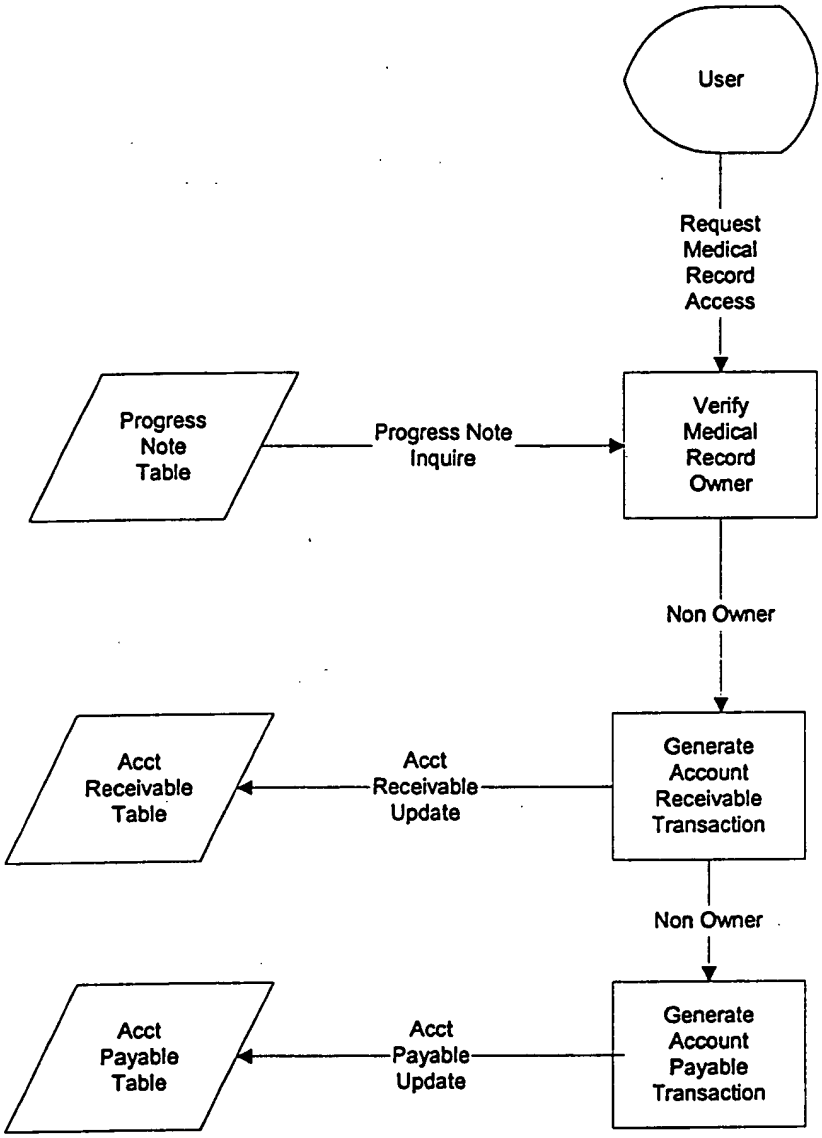
CONFIDENTIAL


Maintain
Pharmaceutical Company
Pharmaceutical Company
Pharmaceutical Company Information
Update Pharmaceutical Company Information
Pharmaceut ical Company Table
Pharmaceutical Update
Pharmaceutical Inquire
Address Information
Contact Information
Address Table
Address Inquire
Update Address Information
Update Contact Information
Contact Update
Contact Table
Address Update
Address Type Inquire
Contact Inquire
Contact Type Inquire
Address Type Table
Contact Type Table

Validate Progress Note

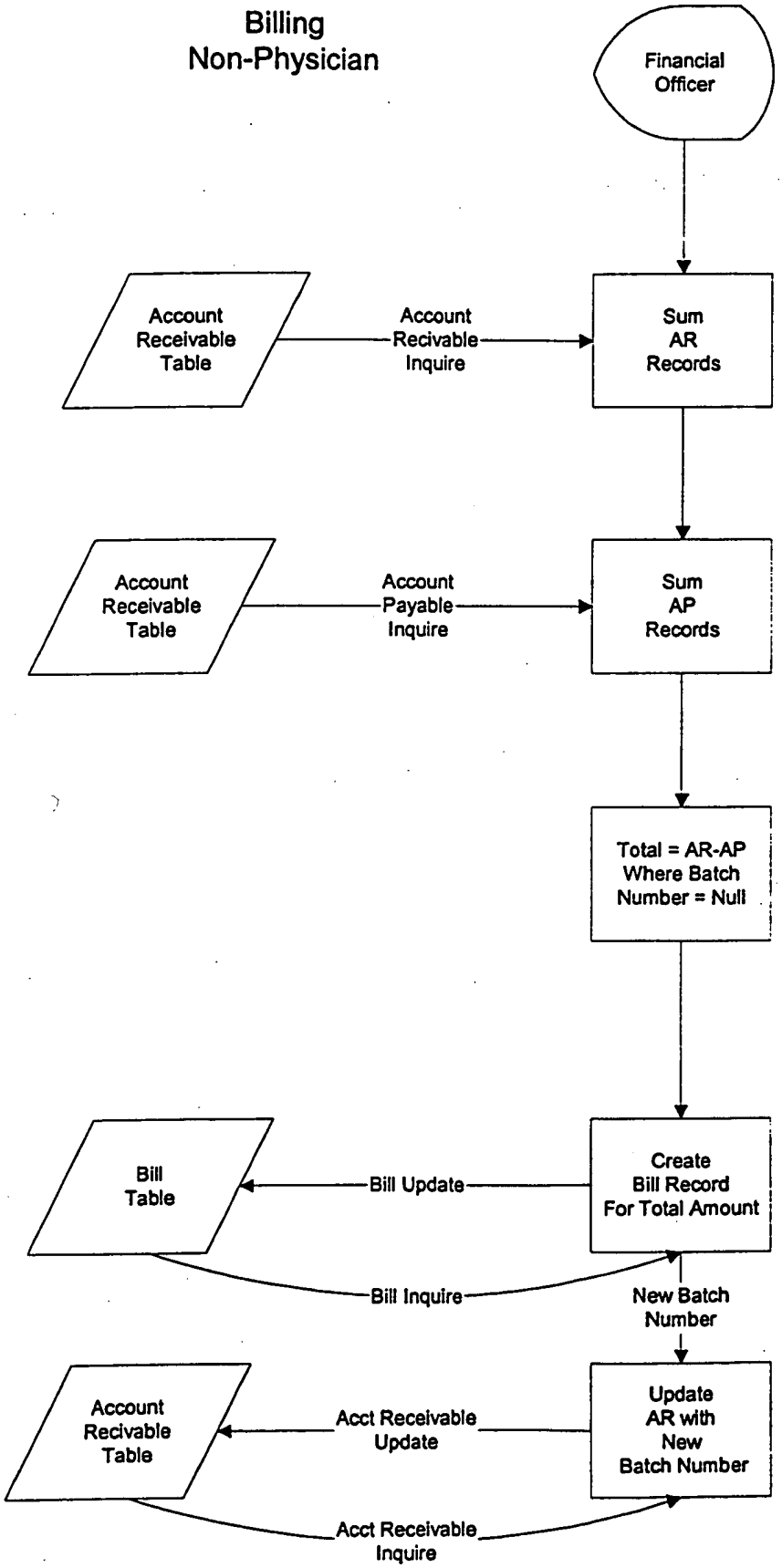


Record Access

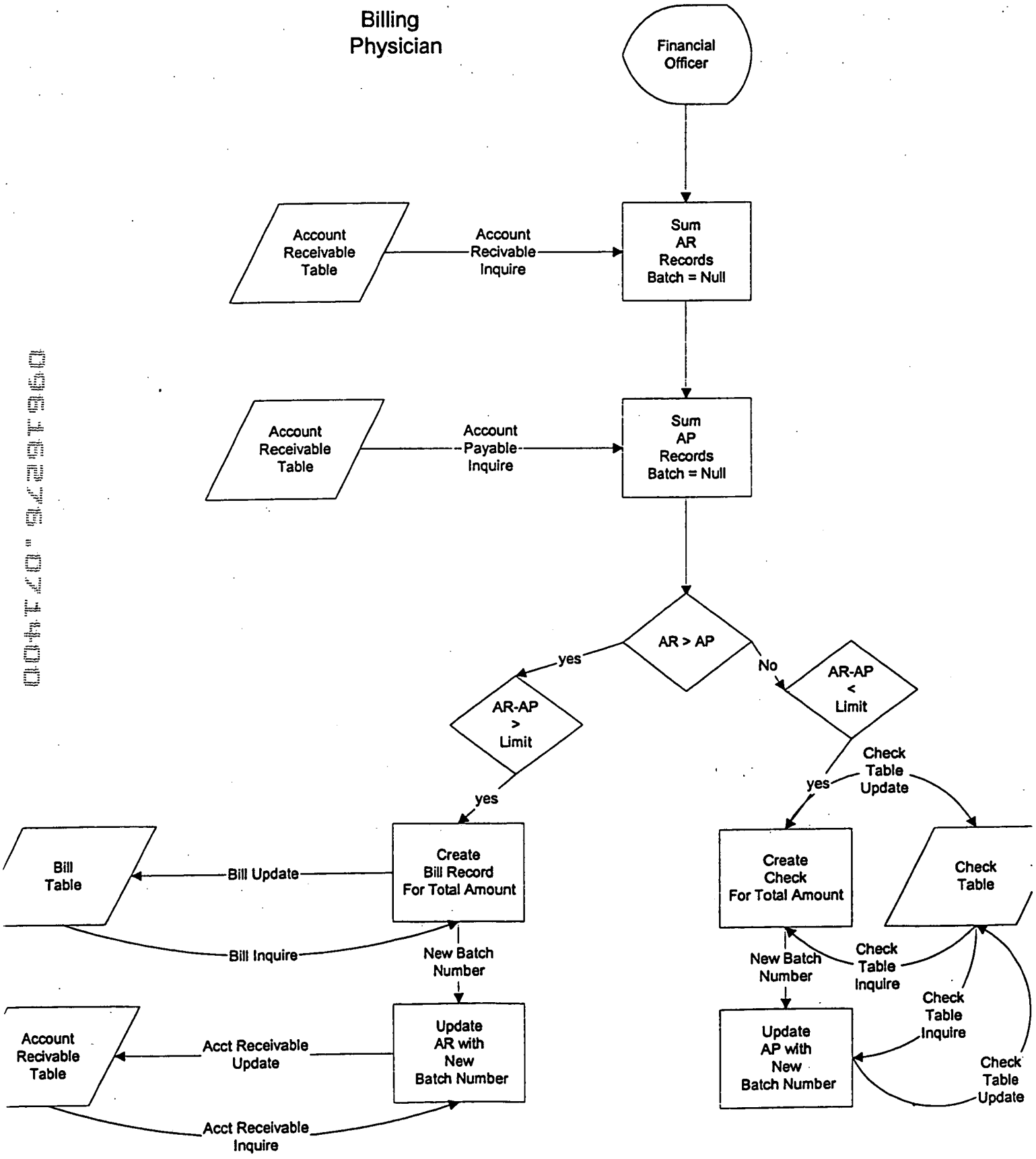


CONFIDENTIAL

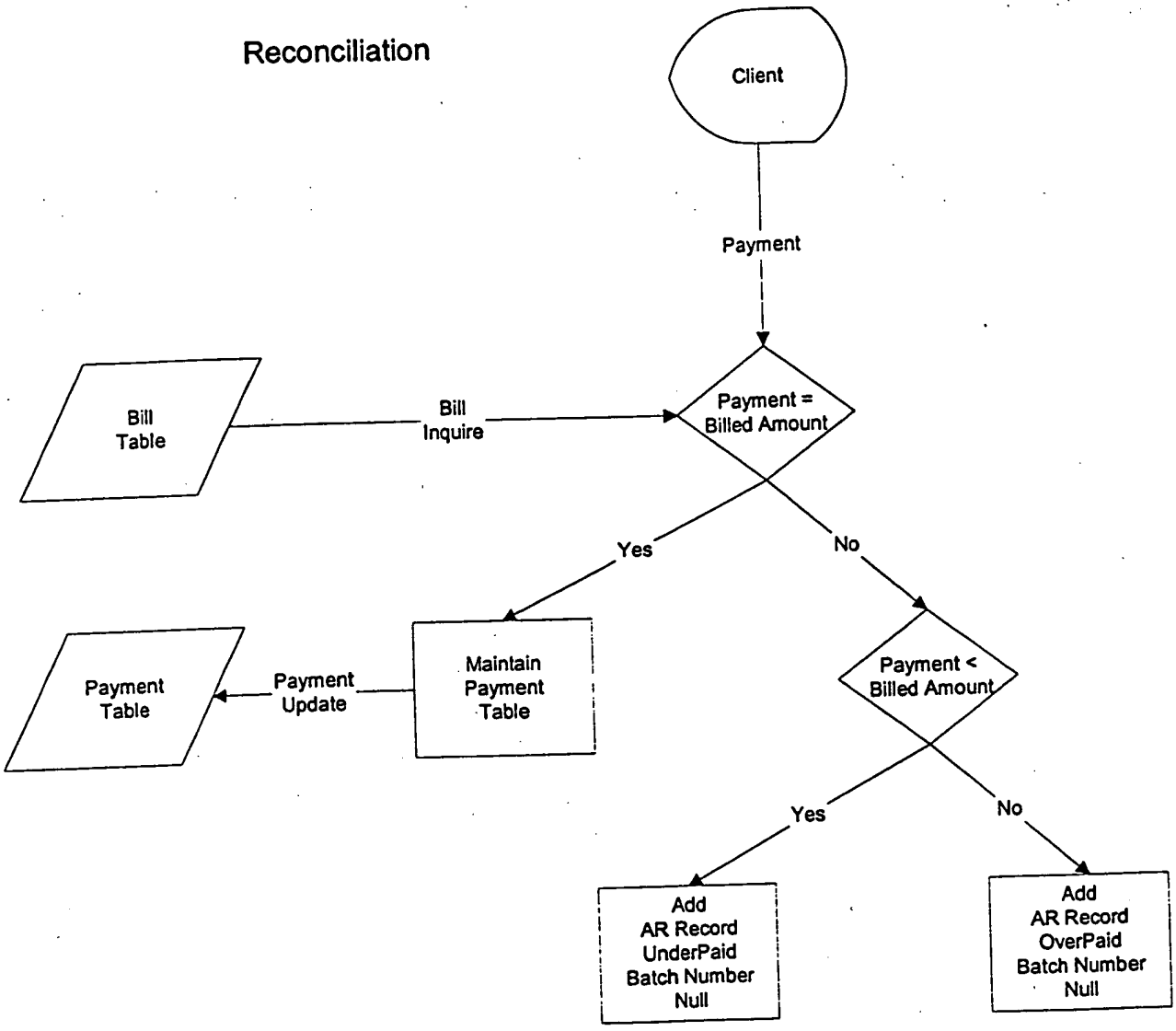
Billing
Non-Physician



MEDISYN


Billing Physician
Financial Officer
Account Receivable Table
Account Recivable Inquire
Sum AR Records Batch = Null
Account Receivable Table
Account Payable Inquire
Sum AP Records Batch = Null
AR > AP
yes
No
AR-AP > Limit
AR-AP < Limit
yes
yes
Check Table Update
Bill Table
Bill Update
Create Bill Record For Total Amount
Bill Inquire
New Batch Number
Create Check For Total Amount
Check Table
New Batch Number
Check Table Inquire
Check Table Inquire
Account Recivable Table
Acct Receivable Update
Update AR with New Batch Number
Acct Receivable Inquire
Update AP with New Batch Number
Check Table Update

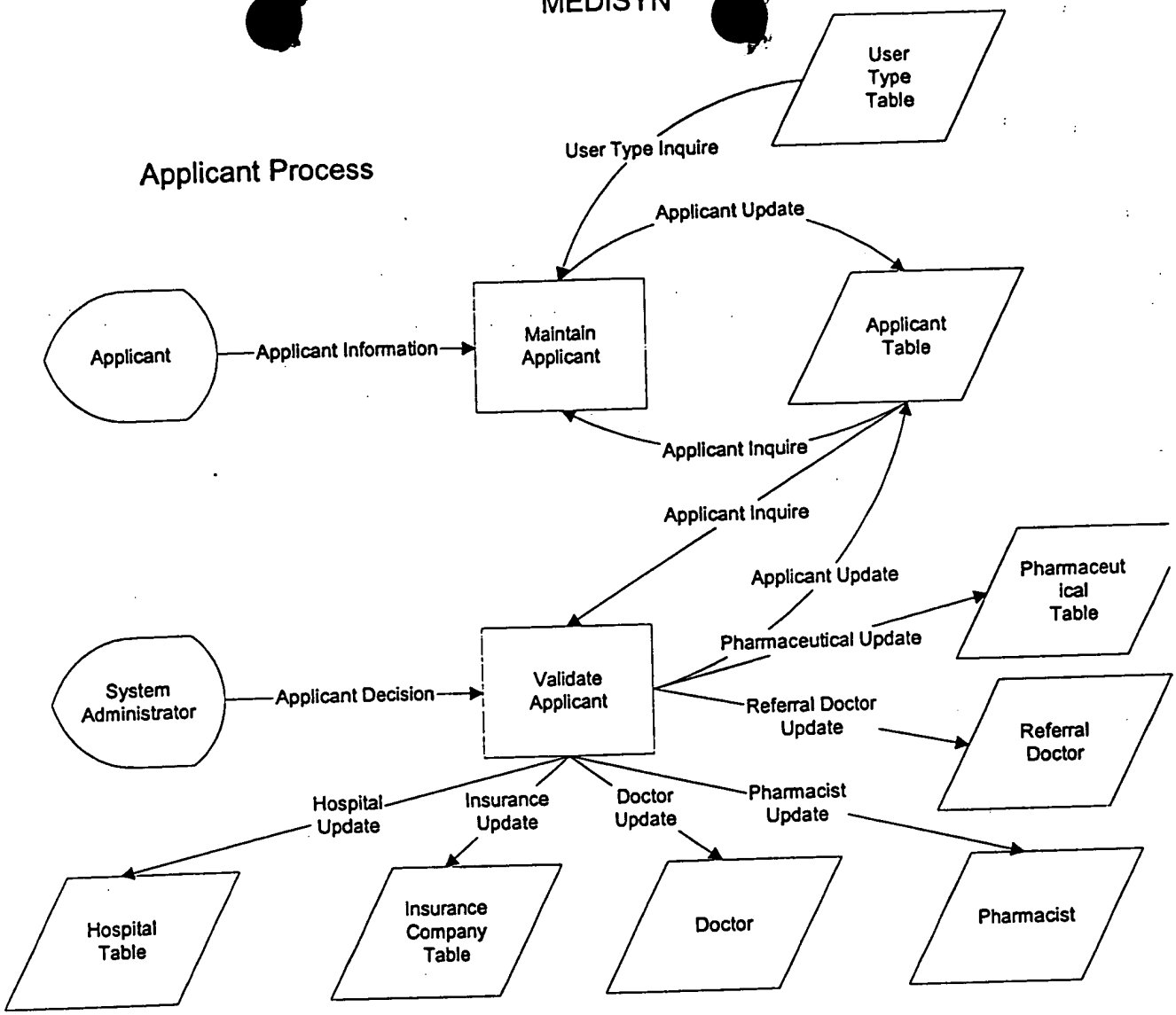
Reconciliation



Confidential - Do Not Copy

MEDISYN

Applicant Process



This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☑ BLACK BORDERS
- ☑ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☑ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☑ GRAY SCALE DOCUMENTS
- ☑ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.
As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.